GENERATIVE ADVERSARIAL NETWORK-BASED VISUAL-AWARE INTERACTIVE FASHION DESIGN FRAMEWORK

By

Ashenafi Workie Dessalgn

A Thesis Submitted to Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Office of Graduate Studies

Adama Science and Technology University

October 2020

Adama, Ethiopia

GENERATIVE ADVERSARIAL NETWORK-BASED VISUAL-AWARE INTERACTIVE FASHION DESIGN FRAMEWORK

By

Ashenafi Workie Dessalgn

Advisor: Prof. Yun Koo Chung (Ph.D.)

A Thesis Submitted to Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Office of Graduate Studies

Adama Science and Technology University

October 2020

Adama, Ethiopia

APPROVAL PAGE

The author, the undersigned, members of the Board of Examiners of the final open defense by "Ashenafi Workie Dessalgn" have read and evaluated his thesis entitled "GENERATIVE ADVERSARIAL NETWORK-BASED VISUAL-AWARE INTERACTIVE FASHION DESIGN FRAMEWORK" and examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirement of the Degree of Masters in Computer Science and Engineering.

Name	Signature	Date
Ashenafi Workie Dessalgn		
Name of the Student		
Prof. Yun Koo Chung		
Advisor		
External Examiner		
Internal Examiner		
Chair Person		
Head of Department		
School Dean		
Post Graduate Dean		

DECLARATION

I hereby declare that this MSc. a thesis is my original work and has not been presented for a degree in any other university, and all sources of material used for this thesis have been duly acknowledged.

Name: Ashenafi Workie
Signature: _____

This MSc. thesis has been submitted for examination with my approval as a thesis advisor.

Name: Yun Koo Chung (Ph.D.)

Signature: _____

Date of submission: _____

DEDICATION

To those who lost their lives through the pandemic of COVID-19 and Mr. Baye Alebachew, the former dean of Engineering School at Wollo University, who died in a sadden heart attack.

ACKNOWLEDGMENT

First, I would like to thank Almighty God and his mother, Saint Virgin Mary, for helping me reach this milestone after so many ups and downs.

My special deepest gratitude and heartfelt thanks go to my advisor and computer vision special interest group Leader Prof. Yun Koo Chung (Ph.D.) for his persistent guidance, valuable support, and supervision form proposal to the completion of this thesis.

My Second special thanks go to Dr. Mesfin Abebe (Ph.D.), who is a postgraduate program coordinator, Dr. Bahiru (Ph.D.), and Dr. Rejash Sharma (Ph.D.) for their voluntary evaluation of my work during the first progress presentation.

I would like to pay special thanks to Dr. Worku Jifar (Ph.D.), Dr. Teshome Megerssa, Mr. Anteneh Tilaye (MSc.), Minyamer Gelaw (MSc.), and Minilik Sahilu for there interesting comments and kindness support by reading my thesis. I want to thank Mr. Cedric Oeldorf, research staff at Maastricht University, the Netherlands, for his help regarding labeling my dataset.

Finally, my thanks go to Dagmawit Semere and Alemitu Demilie for their effort during dataset collection and annotation. I also thanks computer vision program staff and postgraduate students, friends, and parents for their relentless support and advice in my life. May Almighty God always Keep safe and bless them through their life.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF SAMPLE CODES

LIST OF ACRONYMS

AdaIN	Adaptive instance normalization
CADD	Computer-Aided Design and Drafting
CAGAN	Conditional Analogy Generative Adversarial Network
CGAN	Conditional Generative Adversarial Network
CycleGAN	Cyclic consistent based Generative Adversarial Network
Colab	Collaborate Laboratory
CUDA	CUDA
DLib	Deep Learning Library
et al.	and others
FashionGAN	Fashion-based Generated Adversarial Network
GAN	Generative Adversarial Network
MINIST	Modified National Institute of Standards and Technology database.
Pix2pix	Pixel to pixel
Poly-GAN:	Poly generative adversarial network
ProGAN	Progressive Generative Adversarial Network
RAM	Random Access Memory
SIG	Special Interest Group
SRGAN	Super-resolution Generated Adversarial Network
StyleGAN	Style-based Generated Adversarial Network
TextureGAN	Texture-based Generative Adversarial Network

LIST OF ABBREVIATIONS

3D	3 Dimensions
API	Application Programming Interface
AI	Artificial intelligence
DL	Deep Learning
ML	Machine Learning
API	Application Programming Interface
C^{++}	C-plus plus programming language
CNN	Convolutional Neural network
CPU	Central Processing Unit
CV	Computer Vision
DCN	Deconvolutional Generative Adversarial Network
FID	Frechet Inception Distance
GPU	Graphical Processing Unit
CNTK	Microsoft Cognitive Toolkit
MS	Microsoft
MSI:	Micro-Star International Co., Ltd
OS	Operating Systems
PPL	Perceptual Path Length
REQ	Research Question
RNN	Recurrent Neural Network
RTX	Ray Tracing
SSD	Solid State Drive
TF-Records	Tensorflow Records
TPU	Tensor Processing Unit
URL	Uniformly Resource Location
VAE	Variational Auto Encoder
SR,LR,HR	Super, Lower and High resolution

LIST OF SYMBOLS

c or y	Condtional inputs
c	Color
D	Discrimnator
D(x)	Discriminator function
E_c	Estimated color
G	Generator
G(c,t,s)	Generator on a color,texture and shape
G(x)	Generator function
L(x)	Loss function
L_g	Loss of generator on a condtion
L_w	Loss of wassertain
P_g	Propablity of generated
P^c_g	Propaplity of enerator color
P_r	Probality of real
s	Shape
t	Texture
Tr	Trace matrix for texture
X_r , μ_r	Real image distances and it's mean respectively
X_g , μ_g	Generated image distances and it's mean respectively
$\check{Z}$	Generator at noise
σ_x	Variance of the input content
σ_y	Variance of the style
λ	Weights

ABSTRACT

Fashion image generation is the task of generating realistic fashion images from a real dataset distribution. Due to the subjectivity of design, fashion industries have always been striving to meet customers' needs. Although image generation techniques have become more advanced through time, the results are prone to visual-inconsistencies, enormous artifacts, uncontrolled generation, and poor quality at large. This study aims to scale up the image generation process by integrating in the GAN for multiple fashion attributes such as color, shape, and texture to existing architectures. To accomplish this, 12000 Ethiopian fashion images were collected from different sources. As deep learning is a data-intensive approach, image augmentation was used to enlarge the dataset to 90,000. Standard preprocessing was applied to normalize inputs to a common scale, compute average color, create a binary segmented mask, and label the dataset. The conditional inputs added in the proposed architecture are an average color, a segmented binary mask, and a 512 texture dimension organized as Tensorflow records fed to the existing progressive growing GAN generators. The discriminator was assigned to estimate the average color and classify the generated images. Besides, two experiments were done using the same dataset and training configuration namely StyleGAN and it's conditional version. Improved results were obtained from the three experiments with the evaluation metrics Frechet inception distance, and perceptual path length 41 and 1500, respectively. Moreover, the human evaluation user study was perfomed to assess the user capability on identifying real and generated images and to examine the closeness of these images by using Google forms in both paired and unpaired evaluation settings with the confusion rates of 46% and 47.6% respectively. In general, the performance evalution of implemented conditional progressive growing generative adversarial network along with multiple conditional inputs shows an improved results were achived. As a result, fashion design could be generated using such a GAN methods.

***Keywords**: Interactive fashion design, GAN, conditional ProGAN, StyleGAN, conditional styleGAN, and Texture GAN.*

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

Nowadays fashion product holds a global market volume of investments with multi-billion-dollar economic revenue in Ethiopia and worldwide at a large [1]. This vast volume market needs a series of fashion design principles and production techniques to meet customer needs [2]. Before the mid-19th century, clothing materials were handmade and the production mechanism was slow. The coming of new technologies such as a sewing machine followed by modern factory has led to massive production systems. Since then, fashion industries have produced massive items in standard sizes and sold at a fixed cost in the global market [3][[4], applied in esthetics design, and increasing the beauty of fashion items in the interest of users [5] [6] [7].

Fashion design and brand systems is an important factor in all production systems across different domains [8]. In practical design and manufacturing processes meeting the interest of users is difficult due to the tag subjectivity and semantic complexity of features for the massive production of fashion items [9]. The main problem becomes even more challenging in considering a user's personalized fashion styles that are commonly based on the interest of the designer. To handle such a problem in the past industries were used computer-aided design and drafting [10] to simplify the design process is using. Using CADD[1] fashion software didn't take long because of the massive production of items needs manual works for design even though it can be applied for initial drawing [11].

After the coming of artificial intelligence, user preference selling platforms were introduced to billon fashion users. The effectiveness of such a platform not doubtful but a further research investigation is required to create a user preference-based design. This can extend to fill the gaps of the common design strategy which is the use of software tool based fashion design. This generic problem requires a lot of research to bring an alternate solution by which the investigation lies in the major categories classical machine learning methods.

[1] CADD is a drafting and prototyping mechanism for any design and documentation

Applying machine learning on fashion e-commerce platforms may ranging from design ease of apparel shopping through search items personalized outfit styles. [12] [4].The feature that can further enhance customer experience is to visualize an outfit on a human body [13] [14]. To do so, this needs to target an agile design that means through the involvement of human experience using machine learning.

Deep learning has taken on a wide range of challenges in the last several years, and now it shows good remarkable results with better models [15]. After coming generative adversarial network[2] [16] the image generation task becomes easy. It consists of two competing neural networks called generators that generate an example from latent space or Gaussian noise whereas, the discriminator network classifies as the real and fake. This research intentionally used a generative adversarial network to solve an existing problem instated below.

In the past, some research was done and showed that the role of the generative adversarial network in design fashion items [17]. The previous works have researched the common domain-based issues in GAN and fashion-based constraints stated by the researchers. A FashionGAN [18] proposed to display fashion garment items by submitted a novel method condition generative adversarial network [19], which is initially derived from condition generative adversarial network [16] . It is an improvement of vanilla by adding extra information label, class, and text to control the output. These methods describe the basic attributes of the input fashion items [20]. The downside of this research it works for single-color and regular, which cannot map an irregular one with multiple colors, style, and pattern. Again, its architecture and the implementation are too complex though, it requires further investigation stated in the paper.

The proposed research is entitled "GAN Based Visual aware interactive fashion design framework ", which objectively conditioned the fashion images generation in texture, shape, and color. Again reducing the color mapping and artifact problems also to achieve better quality image by integrating common fashion attributes as conditional inputs for the existing progressive growing generative adversarial network (ProGAN) architectures. ProGAN shows impressive results, but it luck conditional architecture. This study plan to add such condition and implement in Ethiopian fashion items were collected from various sources.

[2] A Generative adversarial network (GAN) is a deep neural network architecture with two neural networks: a generator network and a discriminator network.

An image augmentation techniques were used to enlarge the dataset to 90,000. The proposed architecture merges an average color and binary mask along with the texture dimension in architecture discussed in Chapter Four. Moreover, additional experiments were done, namely style-based generative adversarial network [21] [22].

1.2. The Motivation of the Study

A generative adversarial network can create a realistic image from Gaussian noise through guided by real data distribution. The research was done GAN using public datasets. The motivation to study on the generative adversarial network using the Ethiopian fashion dataset. But unfortunately, an Ethiopian fashion dataset was not found anywhere. As a result, the researcher is motivated to collect those fashion images from various sources. Again the existing GAN-based fashion generation methods have visual bluer artifacts and consistency problems.

The proposed research intended to generate fashion images by adding multiple conditioning local structure patterns of style and colors on Ethiopian fashion items. So that this study can contribute in many aspects shown as follows:

- The proposed framework can generate virtual garment images without any operations and also limited control by conditioning the inputs.
- The proposed framework enabled control and condition the synthesis of the generation and can produce a super-resolution result that is high-quality resolution images across the iteration of training
- Develop architecture and simultaneously evaluate the results generated from the proposed and other common methods.

1.3. Statement of the Problem

The statments of the problems in this study could be considerd as a couple of issues. The first is from the domain of interactive fashion design point of view, where as second one is on generative adverserial techniques, and an improving mecchanisms on architecture that used to enhances the image generation quality and how add mutiple conditional inputs to keep the interest of users, since fashion is a challenging task by itself full of complexity attributes and features [20].

Since fashion design is exposed to subjectivity for individuals in personal style [21][22] outfit, color, and texture, and shape choice that brings user dissatisfaction. After coming off the generative design model, the image generation is on a good track of creating realistic fake contents from large probabilistic input distribution. In this study, a GAN based visual aware fashion design methods are proposed and developed to reduce the visual artifacts and inconsistency problems on attributes of generated results such as shape, texture, color, and patterns. An architecture is developed to in advance multiple conditions such as color, texture, and shape by using a new Ethiopian fashion dataset ,which has several complex texture,color,shape and patterns. An additional task that this study performs on collecting fashion images from different source and after severial preprocessing make ready it for generatve adverserail training.

1.4. Research Questions

This study intended to answer the following three significant research question.

1. How to integrate multiple conditional inputs and incorporate them into the image generation process?
2. How to measure the quality of multiple conditional fashions generated model and images?
3. How to evaluate the effectiveness of the proposed method with existing methods in terms of quality of generated images and model with time taken to evaluate it?

1.5. Objectives of the Study

1.5.1. General Objective

The general objective of the study is to design and develop an interactive fashion design framework using generative adversarial networks.

1.5.2. Specific Objectives

The following are specific objectives under the proposed research to achieve the main objective.

- To develop an architecture by integrate conditional input with it.

- To implement the architecture on a progressive grown generative adversarial network.
- To train the proposed method with collected personal dataset.
- To compare the proposed method with styleGAN and conditional StyleGAN architectures in terms result quality and time taken to evaluate the model.
- To evaluate the model using a standard evaluation metrics Frechet inception score, perceptual path length.
- To evaluate the generated images using human evaluation through questionary.

1.6. Scope and Limitation of the Study

1.6.1. Scope of the Study

This study focuses on developing an architecture for multi-conditional fashion image generation. The implementation of the generative adversarial based architecture is to control the fashion generation tasks. Evaluating the quality of generated results also intentionally done. The proposed research conditioning is based on additional information. As a result, the scope ranges from collecting datasets to developing interactive fashion design and fashion image generation to evaluate the results in contrast with others previously done approaches.

1.6.2. Limitation of the Study

This study is limited to generate images based on multiple conditions. The limitation of this study comes from constraints such as constraints on a tight schedule, Luck of 3D datasets, and lack of computational resources. Again from the conditional information used in the study, complex patterns and interactivity was not addressed..

1.7. Contribution and Beneficiaries of the Study

1.7.1. Contribution the Study

The contribution of this thesis is listed as follows

- Improving quality of generated images by using progressive grown training.
- Improve visual inconsistency compared with existing traditional GAN training methods.

- Since the method is conditional progressive growing with an improved architecture along with multiple disentangle conditional inputs.
- Proposing a fusing of an algorithm with multiple feature conditional inputs for texture, binary mask, and the average or dominant color.
- Collecting anew Ethiopian dataset and open for the research community in Github.

1.7.2. Beneficiaries of the Study

The main beneficiaries of this study can be applied in different domains such as

- Design for Manufacturing: it can be used as a creative tool to create prototypes quickly.
- Style transfer: generating multiple unseen cloth fashions with user preference [23] and clip summarization [24], and painting arts.
- Photo editing and image enhancement: improve quality and control the synthesis color and shapes.
- Fashion Image search engine: create a large scale fashion search image engine [25].
- Fashion Recommendation: show a visual and semantic feature of cloth for user preference [26].

1.8. Organization of the Thesis

This thesis is organized under seven chapters as follows.

Chapter 1: describes the background, motivation, statement of the problem, objectives, scope, and limitation, introduction to the methodology, and the contribution of research.

Chapter 2: discusses the background literature and related works regarding interactive fashion design.

Chapter 3: presents the methods from data collection, preprocessing, and annotation techniques and evaluation mechanisms for results and models.

Chapter 4: covers the proposed solution in a detailed flow chart, block diagrams, and pseudo-code in both training and testing algorithms with the equivalent mathematical expression.

Chapter 5: discusses the implementation step of the proposed solution, implementation sample source code of the proposed solution.

Chapter 6: presents the proposed solution results and compares the new result of the proposed solution with the previous solution and elaborates on the meaning of the results.

Chapter 7: summarizes and concludes the work along with the result and it explores the open issues in the research.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1. Introduction

Computer vision is an entire field of study which intentionally focuses on how machines or computers make an understanding of digital images. It tries to automate a human being's visual system task systematically, after the advancement of classical computer vision into the advanced machine learning techniques known as deep learning. Classical computer vision methods work for both feature processing and feature extraction at a time manually. But in terms, deep learning is done simultaneously without explicitly doing feature extraction. In the last two-decade deep understanding has shown remarkable success in multiple real-world problem classification and recognition.

Nowadays, applying deep learning to solve problems in art and fashion design attracts attention. These can strictly change the way people interact with fashion. Due to fashion subjectivity and personalization, the e-commerce platforms have shown excellent results in simplifying apparel shopping through search items. Large fashion industries studied multiple related issues as hot research topics for the massive potential of industrial application [27]. For instance, cloth segmentation, recognition, retrieval [3], visual recommendation [4], and image generation [18][28] [29] are the hottest thematic researches areas in fashion development and production right now.

This chapter started by defining fashion design and followed by investigating existing methods of image generation tasks along with limitations. First, this chapter presents a broad view of fashion design in both traditional and modern fashion design concepts. Then secondly, focus on specific design mechanisms to handle modern machine learning those are ranging from classical computer vision approaches to deep learning and generative adversarial along with their limitation [30].

2.2. Fashion Design and Development

Nowadays, the fashion industry is becoming a three trillion-dollar global market devoted to making and selling clothes [20]. This enormous economic significance of garment

production brings subject to rapid changes in fashion, the greater the demand for cheap products of its kind. Fashion design can be defined as in different terminology by different peoples or communities across the world. Apart from this definition, some of them shown as follows.

2.2.1. Definition of Terms

Fashion design can be defined as in different terminology by different peoples or communities across the world. Apart from this definition is shown as follows.

Definition 1: Fashion design is a vital profession and form of art dedicated to creating clothing and another lifestyle accessory [3] [6][31].

Definition 2: It is also an art of applying design, aesthetics, and likely clothing and along with its accessories. Most of the time, the fashion design strategy is influenced by socio-cultural attitudes varying from time to time [7][32].

Definition 3: Fashion is an exciting interplay of art and science - where science (data) informs us as to the customer's choice, art aims to create aesthetics with broad appeal that has relevance to the times we live in [5].

2.2.2. Traditional Fashion Design

The traditional fashion had always been just an appareling need and request for most peoples across Ethiopian and even the world. Over the last century, in our country, there was a unique traditional appareling need for changing substantially to fulfill the society requirements. Interestingly, classic fashion has reached a very competitive to the local market across in tour country. It is a way of fashion design needs a skilled drawing and manual or hand drawing; the definite production mechanism didn't go to industrial instead of using a unique mechanical tool throwing a ring-like loop along the left and right direction called weaving through or via superior pattern design is done manually by hands through sewing by a needle.

Even though the traditional way of fashion design served as the best strategy for producing the right fashion products, many problems have been there. In fashion design, there is user style subjectivity, time-consuming, everything processed manually and challenging to adopt patterns during production the interest of the user due to subjectivity. Figure 2.1 shown below describe the traditional cloth-making manufacturing in Ethiopian [33].

Figure 2.1 Traditional textile manufacture

2.2.3. Modern Fashion Design

The modern fashion design approaches range from using fashion sketching software to applying new artificial intelligence and machine learning approaches to it. In contemporary fashion, garment making is transforming the artistic fashion design into a given pattern in a variety range of sizes [34][35]. The design issue was challenging to consider and appropriately handle the human-body adjustment. The increment or decrement of the weight cannot be scaled or down in a uniform pattern, and drafting styles are the first and most crucial step in dressmaking. Since the designers usually start with a common draft on paper; add styles, features, and color prototypes; revise and refine everything, and as a final point, deliver their design to dressmakers to apply on it. The fashion design, as a significant discipline, needs extensive knowledge in creating fashion styles [2] [36]. Eventually, most designers give great attention to following the current trends that can help them to predict the popularity of fashion in the future.

The modern fashion design is classified into two major groups. These are ready-to-wear and haute couture. The haute couture collection is devoted to individual specific customers within custom-sized that exactly as per the curiosity of the user's order. On the other hand, the ready-to-wear focused on usual sized, not custom made, so they are more appropriate for large production runs for the generic market. Such research must create a better fashion design framework to predict the next famous fashion through bits of help of artificial intelligence technology [37].

2.3. Application of Machine Learning in Fashion Design

Artificial Intelligence had a significant impact on transforming to radical growth in online fashion shopping and item recommendations have shown remarkable success for effective search within the historical customer experience. It can be applied to design fashion [17], especially a new advanced machine learning, so-called deep learning, and generative adversarial networks [16][19]. A machine-assisted design method chains human experience and deep learning methods in a supervised or semi-supervised manner [2]. As illustrated in Figure 2.1 below the hierarchical advancement of artificial intelligence, machine learning applied fashion design. Computer vision is an amazing domain to use such exciting technology, especially deep learning and artificial intelligence. The progress of deep learning had contributed to the computer vision field in a wide range of aspects.

The application of machine learning can be in multiple areas, including fashion. Artificial intelligence used for fashion apparel classification, design recommendation [38], and generating high-resolution fashion images without fashion designers and get inspiration. Now a day fashion industry using technology and replace fashion designers with artificial intelligence as broad science. For instance, Amazon[3] is working on fashion-related in their respective lab through an algorithm. STICH FIX[4] also identifies the potential gaps of an inventory company's such that clothes design and recommendation are based on better fashion design. The other company is Myntra[5] works fashion recommendation for customers currently favorites and popularities of common combination attributes even though such technology used in industries had many issues in design [39].

After coming of machine learning, a fashion business can stay ahead to deliver exactly what customers want when they want it. It assists the customer in easily looking for better, and

[3] Amazon is an American multinational company that focused on product selling, and shipment on eCommerce, digital streaming, cloud computing, and artificial intelligence.

[4] Stitch Fix is an online personal styling service in the United States. It uses recommendation algorithms and data science to personalize clothing items based on size, budget and style.

[5] Online Shopping Site for Fashion & Lifestyle in India. Buy Shoes, Clothing, Accessories, and lifestyle products for women & men.

different products and even it is possible to adapt their preference from experience [29] to multiple areas, including fashion.

2.4. Preprocessing and Feature Extraction Techniques

2.4.1. Preprocessing of the Data

Data preparation is an essential and primary task of computer vision research. Data preparation is done primarily at the top may differ from domain to the domain, even the nature of data used. The most commonly and widely used preprocessing such as normalize, resize brightness adjustments, and others. Preprocessing techniques work with existing images, fixed angle of point, and other consideration and possible to generate and increase the size and efficiency of the dataset using data augmentation techniques. This method is used for improving the available limited data to large, meaningful, and more diverse amounts datasets[2].

2.4.2. Feature Extraction

Feature extraction is an import computer vision techniques to select out relevant features to work with it. Even though it depends on appearance information, geometric information temporal, and spatiotemporal information, from all feature extraction techniques, only discusses some methods that have a high relationship or contribution to the image generation approach to specifically deal with fashion design[35].

2.4.3. Segmentation

Segmentation is a computer vision technique that search starts by over segmenting the image based on the intensity. Various segmentation methods are proposed along with the binary mask segmentation approach [2] [8]. To show the special attributes segmentation used for the consistency of shapes while in embebded mask generation.

2.5. Generative Model Approach's

It is an approach to learn from various data distribution through an unsupervised manner. There are two commonly used generative model approaches, namely generative adversarial networks (GAN) and variational autoencoders (VAE), based on the real data distribution to generate new data variation in the training set in both generative models.

2.5.1. Variational Autoencoder

It is an unsupervised neural network method that complex distribution. Figure 2.3 here below [40] describes the mechanism for maximizing the minimal sure of the data log-likelihood trained to reconstruct the input into an output.

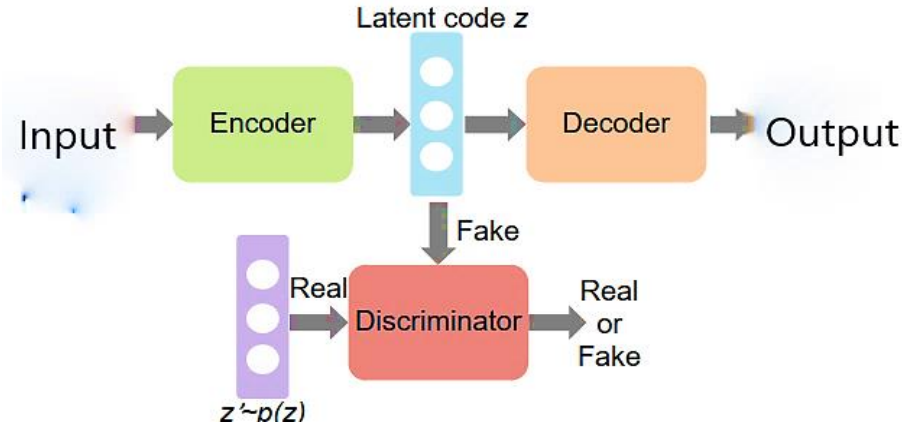


Figure 2.2 Architecture of variational Auto Encoder

2.5.2. Generative Adversarial Network

Generative Adversarial Networks (GANs) [16] are deep neural network models that aim to generate images that look like real data. It consists of three main parts. The generative model to learns and describes data generation in terms of a probabilistic model. The second is the adversarial, the training competing setting in the generative model, whereas the third network used deep neural networks as artificial intelligence (AI) algorithms for training purposes.

Figure 2.4 [41], illustrated the two key components networks of generative adversarial network networks that composed of two networks, discriminator D and generator G. It also shows the backpropagation mechanism for these competing neural network models are the generative model, *G*, and a discriminative model, *D.* So, the discriminator task is to classify the real data comes from training distribution and generated one comes from a generator.

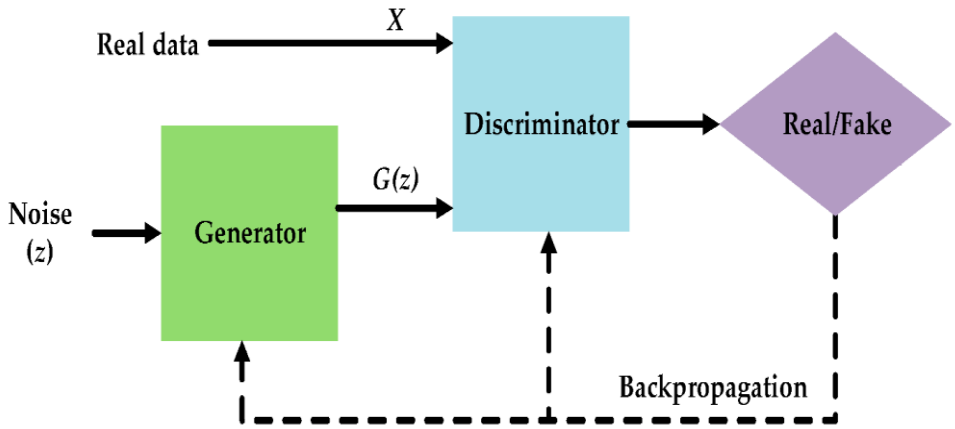


Figure 2.3 GAN architecture and backpropagation mechanism

The generator input synthesized new samples from scratch that means from the random vector (Gaussian noise). At this time, the initial output is also noise. Over time, the generator can generate more "realistic" and looks like the real one. The discriminator is not only to improve result sample quality by giving feed to the generator but also takes samples from both the training data and comparing with the generator's output and classify if they are "real" or "fake" by making it harder for the generator to deceive it through backpropagation. The mathematical Equation (2.1) [16] shows the cost function generative adversarial network[6] distribution. The D(x) discriminator, whereas the D (G (Z)) is the generator network.

(Minimize the generator error by maximize the discriminator learning) = the total real data (x) distributions feed for the discriminator D(x) with its loss but the generated data if from noise so that it is represented in lnoise z for that matter the loss is D (G (Z)).

Loss of GAN= D(x) loss of dicrimnator + D (G (Z)) loss of generator with noise vector. Math maticaly it be expressed as:

$$\min_G \max_D V(D,G) = E_{x\sim pdata(x)}[\log D(x)] + E_{x\sim pdata(x)}[\log(1-D(x))] \qquad (2.1)$$

[6] This equation was take from GAN paper authorized by Ian Good Fellow

2.6. Image to Image Translation

The transformation of an image from the original to some unreal forms while keeping the original structure and its semantics [42][43]. Such computer vision aims to learn the mapping of input and output images. It can be used for many real-world applications, such as style transfer [44], object transfiguration, and photo enhancement. Much research has been done to demonstrate mage translation across multiple domains; these are pix2pix and CycleGAN.

2.6.1. Conditional GAN

GANs can be modified to generate images based on several conditions. The conditional generation is an extension of a primary generative adversarial network through limited conditions, for instance, class information, text descriptions, etc. [19]. CGAN is the same as GAN, except additional details, c condition it. Equation (2.2), the condition used as the auxiliary input layer, as shown in Figure 2.5 here below [45] [46] "c" can be any kind of added y information in Equation(2.2), such as class labels or data from other modalities. The Equation (2.5) here below discusses the cost function after additional information that can condition the nature of fashion image during generation.

$$\min_G \max_D V(D,G) = E_{x\sim pdata(x)}[\log D(x/y)] + E_{x\sim pdata(x)}[\log(1-D(x/y))] \qquad (2.2)$$

Conditional networks use a conditioning variable as the network's input array rather than a random set of values. Once trained, the user can control the generated result by manipulating this variable

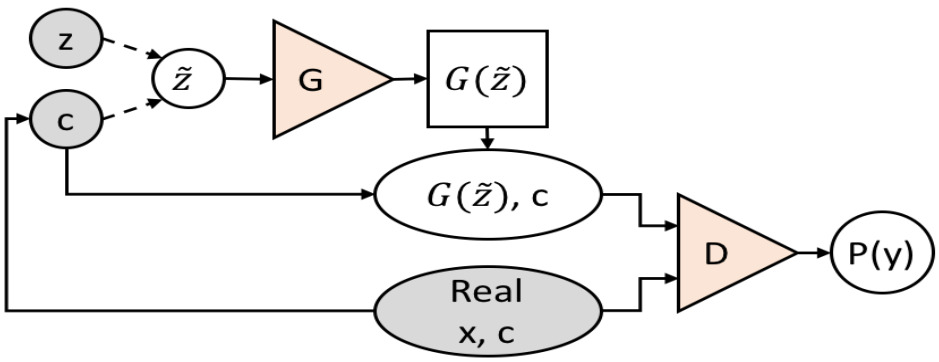


Figure 2.4 Conditional generative adversarial network framework

2.6.2. Pixel to pixel Image to Image Translation

[46] P. Isola et al. proposed a condition-based image-to-image-translation model. The new model (Pix2Pix) not only learned a mapping function but also constructed through a loss function and training mapping. A high-resolution source grid is mapped to a high-resolution target grid. In Figure 2.6 below [46] the *D* learns to categorize the fake (synthetic images generated by the generator) and real {input map, photo} tuples. *G* learns to fool *D*. *G* and *D* can access the input map.

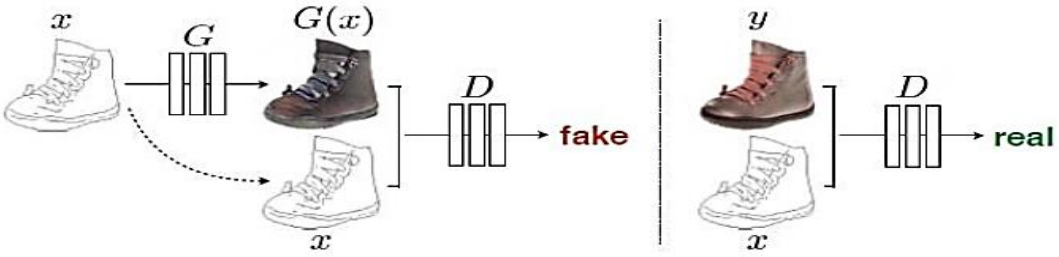


Figure 2.5 Pix2pix conditional image generation

2.6.3. Cycle based Generative Adversarial Network

The main drawback of the Pix2Pix model is that it requires a significant number of labeled image pairs. Later, they improved their method and designed a new model (CycleGAN) [43] to overcome this issue by translating an image from a given source domain to another target domain without having paired examples using a combination of adversarial and cycle-consistent losses. A shown in Figure 2.7 [43] the goal of mapping G: X → Y for images from G(X) is indistinguishable from the Y by calculating the adversarial loss. This mapping highly under-constrained, and this paper also illustrated the inverse mapping such that F: Y → X calculated as F (G(X)) ≈ X and vice versa.

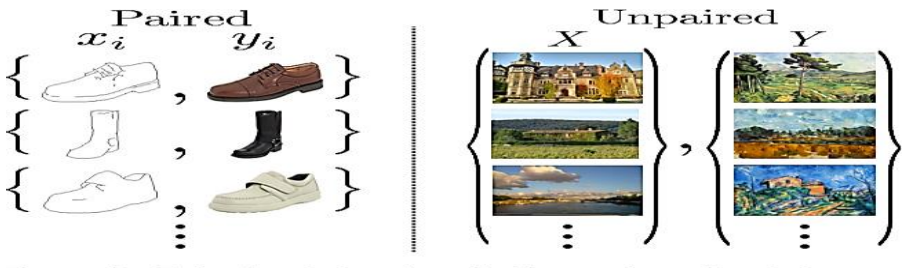


Figure 2.6 Paired training data (left) and Unpaired training (right)

2.6.4. Texture based Generative Adversarial Network

A deep image generation is conditioned by sketch, texture [47], and color [2]. The author tries to addresses a high emphasis on texture first, and then color and drawing also successfully guided the synthesis. The network learns image generation through texture suggestion and experimented with sketch synthesis from real images and texture to generate a super realistic guided by texture.

2.6.5. Super-resolution Generative Adversarial Network

Photorealistic images could be generated from the down-sampled image from low-resolution images that contain some blurry artifacts and better-looking [48] super-resolved images without losing content information illustrated in Figure 2.8 below [49] generates a high-resolution image with 4x up-scaling. It could be used to recover finer details and often generate blurry images.

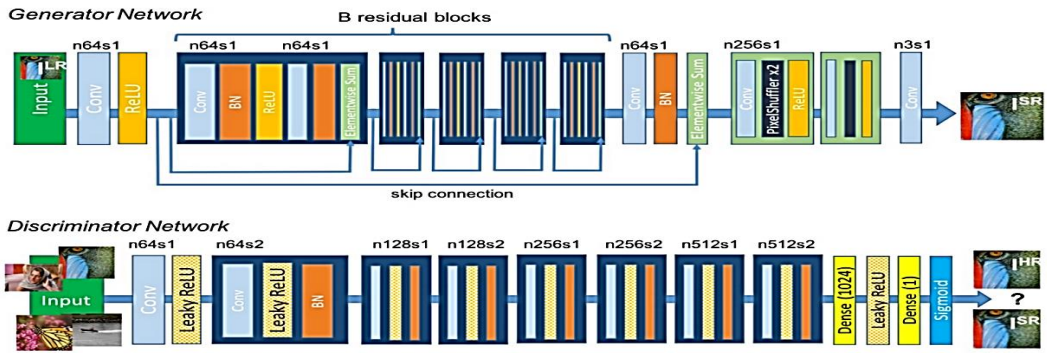


Figure 2.7 Architecture of SRGAN

2.6.6. Progressive Growing Generative Adversarial Network

Generating high-resolution images, a variation of a result, and training instability [50] due to min-baches was the most difficult and challenges in the image generation process. The authors of this paper, proposed a novel generative adversarial network method, so-called Progressive growing GANs. This method can help in training stability and also achieve the best results while generating realistic images and again was to create high-quality or super-resolution images in a progressively growing to the discriminator and generator [29] which initially from low-resolution (4X4) images and then add a new layer which can introduce high-resolution (1024X1024) throughout the entire training settings.

ProGAN can discover low-frequency information and incrementally grows more fined scale high-frequency information. The generator and discriminator network grows in a synchronized manner that means all layer trainable by the training. So that a new layer becomes fade smoothly, as shown in the following Figure 2.9 below [50] the resolution progressively grows from 4X4 incremental grows up to the higher resolution (1024X1024). As the resolution increase gradually, the generator (G) and discriminator (D) simultaneously continue learning.

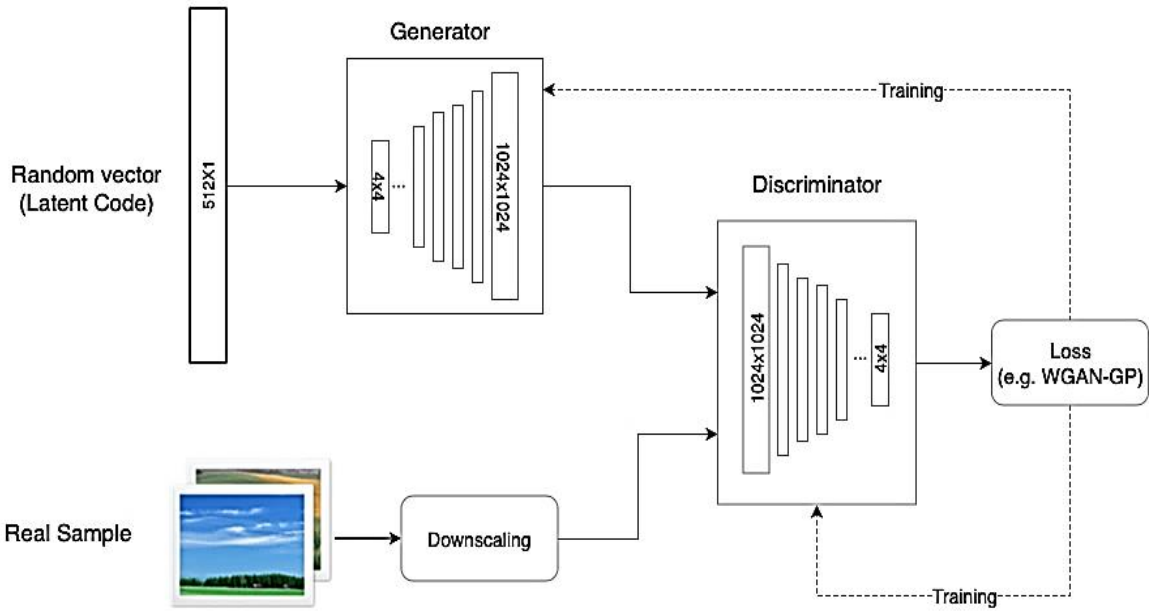


Figure 2.8 Progressive growing GAN architecture

2.6.7. Style-based Generative Adversarial Network

A new proposed new generator that is an extension of progressive generative adversarial with the same main author that improves traditional training [21]. This comes from the fusion of style transfer literature that can start from constant input to control image synthesis to produce high-resolution with better quality. It can generate an image from a static tensor instead of using a stochastically generated latent variable as convolutional GAN, that used as a styled vector through adaptive instance normalization. As illustrated in the following Equation (2.3) [51], the content to be normalized an input x and its style input y by using mean and variance adopted in styleGAN. The mathematical expression look likes

$$AdaIN(x,y) = \sigma_y \frac{(x-\mu_X)}{\sigma_X} + \mu_y \qquad (2.3)$$

Where x is content input, $\sigma(x)$ the variance of input,

$\mu(x)$ mean of input, y is style input, $\sigma(y)$ and it's the variance of style.

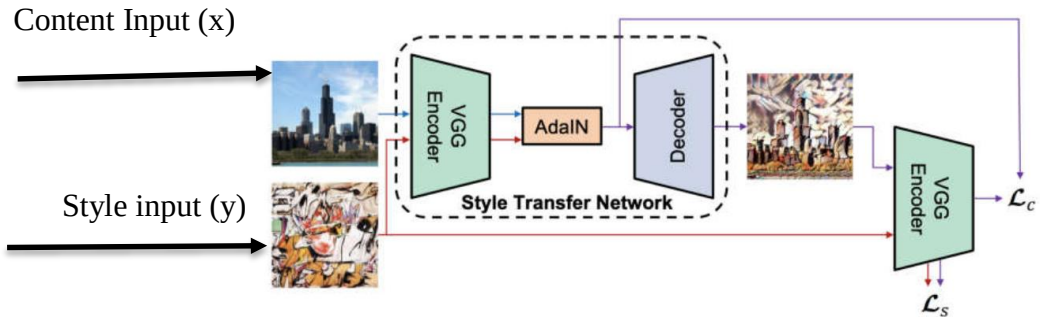


Figure 2.9 Operation of adaptive instance normalization using style transfer

StyleGAN generator assumes images as a collection of "styles", in which each style can control each specific scale. The transformations stated in Equation (2.4) [51] that learn to specialize **w** to and the corresponding styles $y = (ys; yb)$ that control adaptive instance normalization operation. This AdaIn can be used in style based generative adversarial network through the following mathematical expression.

$$AdaIN(x,y) = y_{s,i} \frac{(x-\mu_X)}{\sigma_X} + \mu_y \qquad (2.4)$$

Where feature map x_i is normalized alone style along with style **y**. Authors choose to reuse the word "style" for **y** because similar network architectures are already used for feed-forward style transfer [22].

Most of the time, the generative adversarial network uses a random vector (Z) as input to the generator. In contrast, style GAN ignores the traditional input layer (Z); instead, it easy mapping a network with an intermediate latent space. In mapping network, random vector (z) to the generator is projected for the middle latent space (w) by feeding network mapping. Like ProGAN, the SyleGAN used a progressively grown architecture to attain higher resolution. The critical difference is ProGAN generates an image from a stochastic latent variable whereas the StyleGAN generates an image from a fixed tensor (4X4X512), and the

stochastic latent variable used as a style vector and also the nonlinear transform through a fully connected mapping network is shown as in Figure 2.11 below show [21] compare the progressive growing and style based generative adversarial network and the fully connected networks added for each style[52].

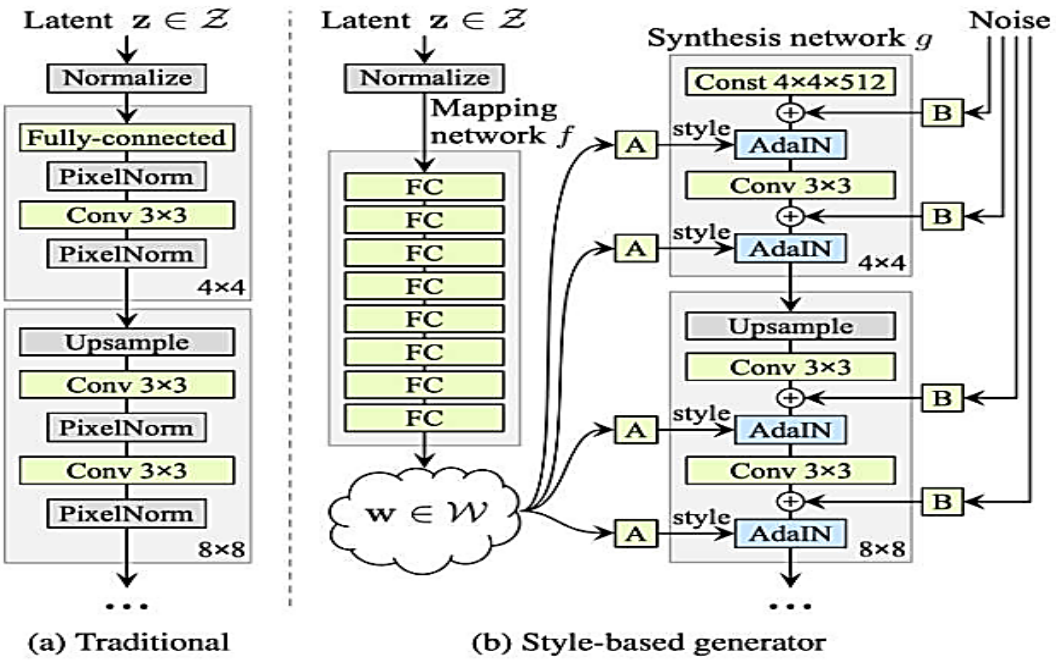


Figure 2.10 Comparison between ProGAN and StyleGAN architecture

The style-based generative adversarial network has unique properties that were not found in proposed earlier. Let us see mention and clearly state how such properties behave in a network since a style-based generator allows to control of the image synthesis in particular scale changes to the style.

- **Style mixing:** StyleGAN can mixing-up the style of multiple images logically. A model by generating two images, that combine A and B then combine low-level features from A and the remaining part of the elements from B.
- **Stochastic variation:** This is conducted by adding a scaled noise to the channel AdaIN module.
- **Mapping Network**: The latent input vector (z) needed to decode by an intermediate style vector (w) where the visual features control generation way in an unsupervised manner.

- **AdaIN module:** Proposed initially for style transfer, where the encoded vector w (style), formed by the mapping network to generated images for each resolution level of the synthesis network.

2.7. Related Work in Fashion Image Generation

Generating a fashion image with a fashion sketch and a fabric image is a one in challenging problem of image-to-image translation [8] [28]. Such a task was done by taking an input of fashion images of people dressed and produce another image that preserves the quality of an image. Image generation tasks are handled by generating images from the Gaussian noise vector called Generative adversarial network (GAN) [16] [19].

Even though in GAN has shown tremendous and successful result had achieved past still in challenging problem for some reason. Here in the following, the author presents a fashion image generation research with their methods used, objective to achieve, and the gap mentioned in the given paper. Image style generation is related to recent artistic style transfer [26] works apply on a generative adversarial network) [16] [19] through instance normalization shows remarkable results [21] [22] [52]

A fashion recommendation that is capable not only of suggesting existing items using CNN and to generate new fashion images through the interest of user preferences using GAN [10]. The gap of this research the quality of the generated images is poor and unable to provide the control of fine-grained styles. A novel approach to new fashions items using generative adversarial learning [13], where the conditional personal image and a corresponding descriptive text to redresses the desired cloth guided by a text keeping unchanged the pose. The gap of research does not assume any constraints or post-processing of the background.

An image synthesis pipeline that can generate garment images based on a fashion sketch and a specified fabric image using CGAN to show images [18]. This paper workable for single-color and regular (like stripe) fabric patterns. But fashionGAN cannot map irregular attributes, the architecture is too complex, workable for simple colors, and doesn't work for irregular patterns, strip and texture may lead to failed results. Multiple inputs were suitable for many tasks, including image stitching, image alignment, and in painting used for architecture for a variety of applications [6] newly proposed a new approach, so-called Poly-GAN. Due to the generic and black box focuses on and less practical many tasks.

A clothing image generation method that has pattern makers [29]. The authors did a user study to investigate the quality of an image in various factors such as epoch and resolution affect the participant's confidence score. The limitation of this paper is the image quality is too low and visual artifacts patterns. Table 2.1 illustrates the summaries of related works. The following are summarized problems that are needed to be addressed in the proposed solution.

- Most of the architecture is too much complex;
- Didn't work for irregular pattern and strip;
- Irregular texture may lead to failed results;
- The generated image is a low-quality and blurred visual artifact on most results ;
- The above issue is fundamental problems that need further investigation or research and motivates us to engage in such a case to fill some of the gaps mentioned.

Table 2.1 below illustrated the summary of related works. From these related papers, this study is going to address only papers with referances [2][10] [18].

Table 2.1 Summary of related works part I

References	***Methodology***	***Objective***	***Limitation***
[10]	CNN and GAN	Prediction and recommendation	Difficult to control style
[13]	Semantic segmentation	Generate image from text	Don't consider the background
[18]	Conditional GAN	Generate garment images	low resolution don't work regular strip
[53]	Poly GAN	Generate image from many inputs	complex architecture due to varieties of input
[2]	ProGAN	Control the color texture and shape	Doesn.t describe the architecture

2.8. Summary of the Chapter

This chapter is organized into main to man two significant parts. The first one presents the theoretical and conceptual clues about common terminology's terms on a domain, methodology definitions in line with equations, and appropriate graphs. The second section related works on image generation algorithms and generation process in line with the method used and gaps to be further researched.

The chapter discusses styles a fashion development consists of both traditional and modern ways of design process or strategies. The contemporary fashion design approach is mainly ranging from using computerized prototyping software tool Like (CADD) to using new merely highly impressive techniques, so-called artificial intelligence. Machine learning can learn from experience by taking examples and produces a program that does the job. Deep learning is an advanced machine learning technique where multiple abstract layers are communicating with each other.

Generative Adversarial Network (GANs) is an advanced branch of deep learning models, consist of a generator and a discriminator which are pitched against each other. Where, The Generator produces fake realistic images from the distribution while the discriminator tries to differentiate and also classify between the generated image (fake or zero) and the real (one) image from the dataset using backpropagation.

Even though the proposed study reduced some of the common issues of generative adversarial network still there are an issue were open for futher investigation. These are training instability (varation of the discriminator and generator learning) , mode collapse (diminvisual in to single output), inconsistency (inconsistent color and shapes in for asingle generated images) and artifacts (unnnessary visual flickers or pixels) and consistency problems. The researcher believe that this open question will solve in future investigations.

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1. Overview of Methodology

This chapter explains the methodology used in conducting the study GAN based visual aware interactive fashion design framework in a net shell. This section describes the data collection, pre-processing and augmentation techniques, and mechanism in line with the research topic with a generative adversarial network. Lastly, this chapter stated the software and hardware tools used in the successful accomplishment of research. It ends by explained the evaluation methods and metrics to assess the result of the study.

1. Collecting the dataset;
2. Preparing and augmenting the image dataset;
3. Labeling the image dataset;
4. Training.

Figure 3.1 shows that the block diagram for building own dataset but internally, it consists of tasks such as collecting from various source, enlarging through augmentation, labeling and annotate create a binary mask, and collectively creating Tensorflow records and feed for training. The dataset building techniques consists of three main steps, as shown as follows 1, 2, and 3, whereas 4 and 5 are the task done after building the dataset.

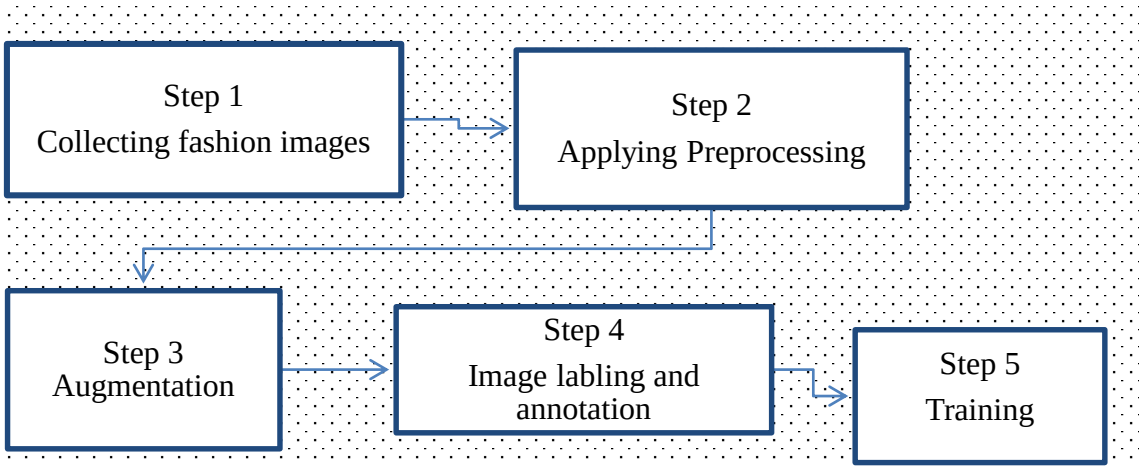


Figure 3.1 Process of building Ethiopian fashion dataset

3.2. Dataset Collection

The main goal of this research is to design and develop a generative adversarial network that works in newly dataset (personal dataset). The proposed architecture objectively must be trained by using a new personal dataset. This is supposed to be done by collecting Ethiopian fashion items images. Publicly available datasets were not used for the following reasons. The first one most public datasets are premium and ready to sell if even and again some of them to release for free but poor in quality. Second, this research is specifically designed for Ethiopian fashion products .Dataset collected from the following sources.

- ***Fashion studio:*** This covers 10 % of the total fashion dataset.
- ***Scrapping websites and social network channels:*** *To* compensate for the shortage of the image dataset and requires looking for a website that has Ethiopian traditional and custom dress bags and other home utility materials. 50 % of the dataset was collected from etsy.com and amazon.com. The other 20% percent of the dataset was collected from the advertisement in social media.
- ***Manual capturing:*** Collecting fashion images manually, having the permission seller at Addis Ababa (Shiromeda) districts.

Table 3.1 Summary of dataset collection sources

S. No	Collected source	Percent (%)	Description	Available
1	Fashion studio	10%	Fashion studios are photo centers, fashion bazaars, and shopping centers.	Shiromeda ,A.A
2	Gaping website	50%	Downloading from e-commerce websites such as etsy.com and amazon.com	Used public domain
3	Social network	20%	Download the advertising fashion images at Facebook group, telegram channel, and Instagram page.	Telegram and publicly aviliable
4	Manual capturing	20%	Manually take photos during cerebration time.	Personal collected
Total aviliable in			Github.com/ashumare/ms_datasets	publicly

3.2.1. Pre-processing Techniques

After collecting the necessary data, the next step is performing pre-processing on the acquired face images to remove some noise, scaling, and filling the content using different techniques.

- ***Normalize image*:** Tensor records created before the data feed to training done, the image resized the image in multiple of 8 significantly 128,256 and 512.Again in daynamic range of images normalizein to [-1,1] and [1,1] .look the detail in Chapter Five.

- ***Remove background:*** Clear background is essential to show the central part of an object in the image. Due to this, all training images had the same white background. This study used aTensorflow background segment API freely aviliable in github.
- ***Images Enhancement:*** Enhance the quality of images using several common image enhancement techniques applied. These enhancement methods could increase efficiency results. Color correction ,brightness control and befor it feeds for training.

3.2.2. Data Augmentation

In the previous chapter the preprocessing and feature extracting approaches are discussed for a large dataset for the performance in the deep learning model. To accomplish this an image augmentations techniques to generate a new dataset from the original ones by adding random jitters. It is essential to produce a data-intensive approach, like deep learning, to feel the need for a large dataset and also the appropriate right way to collect a million images, so it is a better way to increase the dataset size by introducing varieties.

The main operation held during data augmentation can be different according to the application of results. But the standard methods are rotation, brightness, shear, zoom, scale, etc. The following table shows the difference between the dataset before and after augmentation along with cross ponding class label and name. To perform this task there is a Keras[7] library and its ImageDataGenerator class.

[7] Keras is a high-level machine learning framework build on top of Tensorflow.

Table 3.2 Dataset before and after augmentation process

S.No	*Name class*	*Class*	*Before augmentation*	*After augmentation*
1	Bags	1	1131	10978
2	Buildings	2	1055	9,854
3	Cap or hats	3	1049	8,581
4	Coffee cups	4	1294	9,344
5	Color arts	5	1489	8,715
6	Female dress	6	1243	8,280
7	Jackets	7	1109	8,838
8	Male dress	8	1306	9,126
9	Sweaters	9	1115	8,162
10	T-shirts	10	1298	9,052
Total			**12089**	**90,931**

Table 3.2 shown I the above describe the dataset the dataset information and number before and after the augmentation process. To quantify the total average of the augmented is 7.5 times as of the original dataset. But the dtaset class in the above had significant data imbalance . This data imbalance afftected the results to diminish in to larager datasets class and to examine this separate class experments were done. The results were achived in this experments shows significant change than befor.On the other hands the data variation and datset class imbalancements could affects the deep learning model's results.

3.2.3. Dataset Labeling

The proposed require a large set of label data to train the model conditionally, so it needs a class labels for the dataset. These can be done by writing an images name, class number, and color information of over 90931 images to pickle dictionary

3.3. Feature Extraction

This study required a couple of extraction techniques namely average color extraction and binary shape mask generation. The average color summarizes all colors available in the images divided by the summation total of its corresponding RGB section. Binary mask generation is implemented through a set a threshold value and sum up the binarized images.

3.4. Research Tools

The following implementation tools are tools that help during the research. It can be hardware or software used for directly for research implementation as well as reporting purpose.

3.4.1. Hardware Tools

The following Table 3.3 shows the hardware tool used along with its specific function throughout the research.

Table 3.3 Hardware tool used for the implementation of this research

S.no	***Device name***	***Used in the research***
1	Camera	To capture input image for a dataset
2	Hard Disk	Used as storage for large datasets.
3	GPU	To increase the computation and to fasten the training
4	RAM	To accelerate the training process cooperatively with GPU

3.4.2. Software Tools

Research software tools for writing the code, dubbing, visualizing results, and documenting the research process for reporting. Software tools are programming tools and library planned to use as follow:

Anaconda**:** - is an application used to install python programming language with its all modules depending on the python version. It provides a navigator application to view different settings like Jupiter notebook, spider, vs-code, and modules installed in the

environment.

Jupyter Notebook: An interactive web-based application that helps configure, load python API, and write python code.

***Keras* :** is user-friendly, easy extensibility supports modularity**,** and works with Python as a high-level framework or API. It more user-friendly compared with Tensorflow. This**.** It is preferable due to its user-friendly, modular, and extensible work on CPU and GPU though it depends on the task to execute.

Tensorflow*:* - It is a machine learning platform to build a model and deploy in client environments like other real-world applications. It supports machine learning, deep learning, and flexible numerical computation [54].

Python*:* **-** Python programming used to implement and demonstrate this thesis requires many of the drivers used to configure and package to install some device with the computer.

Microsoft Office packages: - MS-office packages are a tool used to write the thesis paper, and Visio is used to design diagrams and flowcharts of the proposed method. Excel to label images with its file name along with its corresponding class label number.

***Mendeley* Desktop**: is a powerful reference manager tool that serves as an academic, social network for referencing similar works.

BibGuru: is also the most excellent reference and citation generator tool that can quickly add sources paper and make citations in IEEE and hundreds of other citation styles.

The following Table 3.4 shows the summary of the software tool used in the research and along with a specific function.

3.5. Evaluation Method

3.5.1. Frechet Inception Distance

The Frechet inception distance measures how much the real and generated images close the each other. In Equation (3.1), shows the Frechet inception distance [55] between fake and real data distribution.

$$\left\|\mu_r - \mu_g\right\|_2^2 + Tr(\varepsilon_r + \varepsilon_g - 2(\varepsilon_r + \varepsilon_g)^{1/2}) \qquad 3.1$$

Where from the real data distributions (Xr) ,(μr mean of the real data and i Σr) and Xg, μg is mean of generated data and Σg) The meaning of the lower FID is the more generated image similar to real images where distance measured activation of the distribution.

3.5.2. Perceptual Path Length

The perceptual path length measures the statistical distribution difference between two consecutive images a time interpolating between two random inputs [56]. It measures the VGG16 embedding features of the images for interpolating between such random inputs.

$$PPL = E\left[\frac{1}{\epsilon^2}\, d(g(lerp\left(f\left(Z1_{,f(Z2)}; t\right)\right), g(lerp\left(f(Z1); (t+\epsilon)\right))\right]$$

Where t added small changes ε1 and ε2 generating an image using latent variables. The shorter path distance show the similarity between t and t + epsilon 1.The image of distance is big distance the dissimilarity between of the two consecutive images. Similar to FID here we use an embedded image distance with trained network called VGG. Since it cannot be determined analytically, the result of performing this calculation on many images and taking the expected value is the PPL value [63]. The lower this value, the more perceptually smooth the latent space is. It is obtained by mixing the two latent variables z_1 and z_2 at the ratio t, and the image generated by the latent variable obtained by mixing the two at the ratio t + ε. g is the Generator, f is a function that converts latent z to style vector W. “lerp” indicate the Linear interpolation. If the data mixed by t and the data mixed by t + ε are close to “perceptual”, take a small value.

3.5.3. Human Evaluation for Paired User Study

The evaluation is done through the website with the URL at human evaluation user study. A paired user study deals with the user assessment evaluation by showing both the real and generated images in the proposed method [18].

3.5.4. Human Evaluation for Unpaired User Study

Unpaired user study deals with the user assessment evaluation by showing only single images to decide what users real or fake.

CHAPTER FOUR

4. PROPOSED GAN-BASED ARCHITECTURE

4.1. Introduction

This chapter discusses the detailed proposed solution architecture with the necessary block diagram, flow charts, and algorithm to well-describe the problem solution. It also gives a brief description of data preprocessing such as the essential steps augmentation, color extraction, binary segmentation, and creating of tensor records and having described such primary task; continuing up to tell the proposed architecture and the along with mathematical expression and essential diagrammatic representation. Figure 4.1 elaborated on the overall architecture during the pre-processing, augmentation, and creating TF-Records.

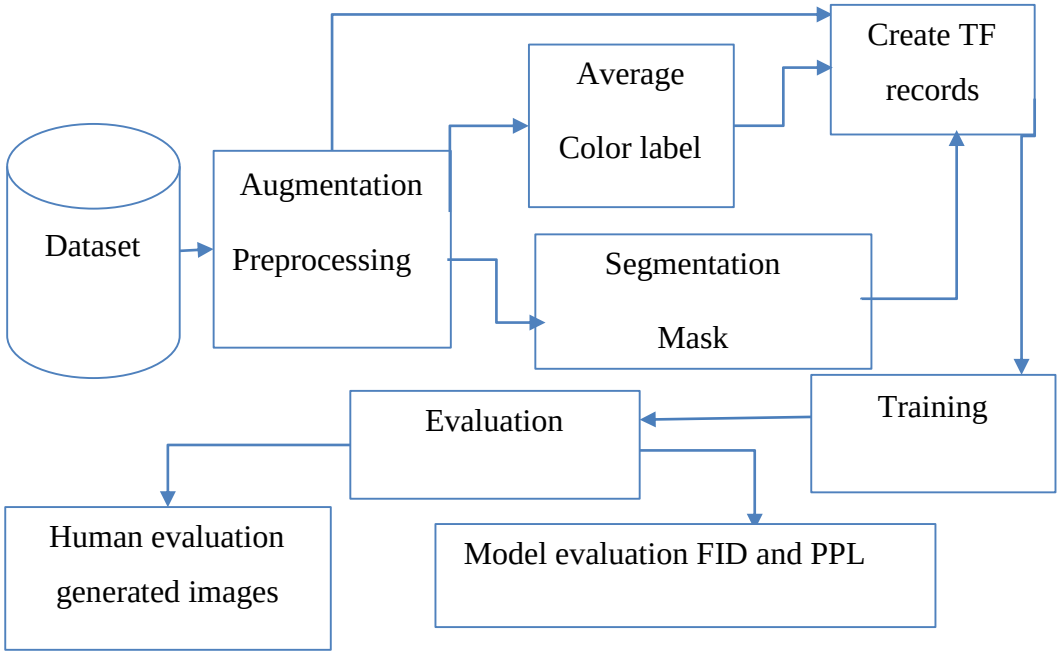


Figure 4.1 The overall block diagram for the proposed GAN-based model

4.2. Pre-processing and Augmentation Techniques

4.2.1. Pre-processing Techniques

Several methods applied for preprocessing images to achieve a better dataset before training to any of the deep learning methods.

A. **Normalization**: is the process of scaling, color bit, and image extension uniform manner such as [0,1] and [-1,1] .

B. **Remove noise**: smoothing an image to remove unnecessary noise on the image, there are various types of noise but , Gaussian noise used to in this study .

C. **Resize image**: Resizing an image on the base of the size which fits the since the study used progressive deep generative algorithm algorithms in multiples o since f 8 such as 64,128,256 , and 512.

4.2.2. Augmentation Techniques

To train deep generative models and algorithms, it requires a large set of datasets with balanced classes. To do so, using an augmentation technique is compulsory. There alternate ways of performing image augmentation, such as shearing, shifting, rotating, cropping to distort the image, and zooming in to a section of the image using a simple machine learning algorithm. As shown in the following Table 4.1discussed an algorithm for the image augmentation techniques.

Table 4.1 Algorthim for Image Augmentation

1. Load the input images to generate others by using the data generator

2. Transform input image through translation or rotation.

3. Take the transformed images to write to disk

4. While the total number of an image need number needed to completed

Transform the input through translation shift to left and right and rotation with o dgree .

Write each augmented result to disk

end

end

Having worked with powerful preprocessing techniques, the second step is making an annotation or labeling it is required. The most important of image labeling along with a critical class number, name, computed colors, and others.

4.2.3. Segmentation Mask Generation

The binary mask is used to control the shape of images during the generation process using. The method used to generate a mask for images by using the threshold [8]value shows as the following Figure 4.2 below, summing summation of the threshold and binarized image. If the pixel value is lower than the threshold value, set the amount to 0. Otherwise, set the maximum value, which is generally 255. There are many deep learning mask generation methods used today but the efficiency is not much as classical computer vision.

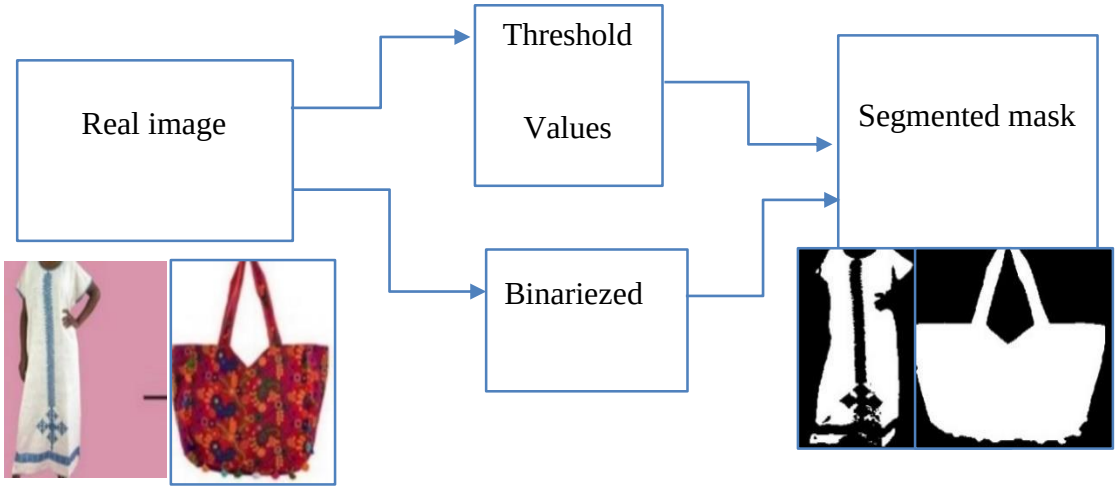


Figure 4.2 Process of the generated segmented mask using

4.2.4. Labeling and Annotation

A) Average color computation and labeling

Average color computation is used to present corresponding color information and data types of real images. Figure 4.3 shown below describe the basic that for average color extraction and labeling process

[8] It is a mechanism to assign pixel values along with based on a comparison of the threshold value

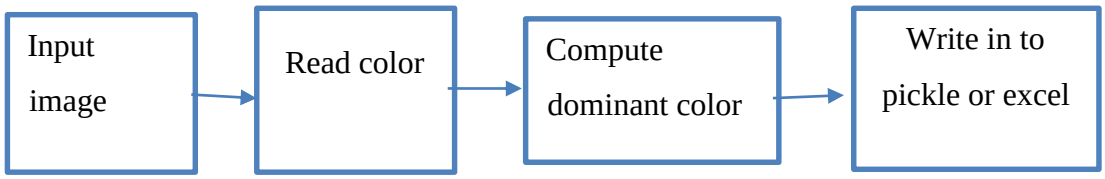


Figure 4.3 Labeled Color information to present the dominant color of each image

The average extracted and written into the pickle dictionary along with name and data types of images. Table 4.2 the detailed algorithm for average color computation with the normalizing [-1,1] to labeling as pickle file.

Table 4.2: Average Color Extraction

1. Load the input image by specifying the directory of the dataset

2. Initialize pickle with a null value

3A. Calculate averages color per row

3D. Read the images and computes the average and dominant color presented in the image

4A. while the total number of image needed to computed averages colors (len(images))

Keep calculates the NumPy averages color per row (repeat step 3A)

Write images name key with values computed averages color write to the pickles

end

end

4.3. Image Generation Process

As explained in chapter two, the image generation tasks were held by using a generative adversarial network [30] [31] and autoencoder [28]. The generative adversarial network is a special subset of deep learning that consisting of two competing networks. This study used the generative adversarial network as a means of generating new fashion images conditioned by multiple information capable of a visual artistic fashion image.

4.3.1. Generator Training

The main goal of the generator network is to generate realistic images that look at the real dataset. Based on these, the generator backpropagates by an update to the generator more realistic to dataset probabilistic distribution. Whenever the generator feeds to the discriminator network, the generator began to generate the output. Table 4.3 shows the generator network training and the backpropagation mechanism to reduce the mistake and also to create a more realistic image based on the calculated difference between real and fake.

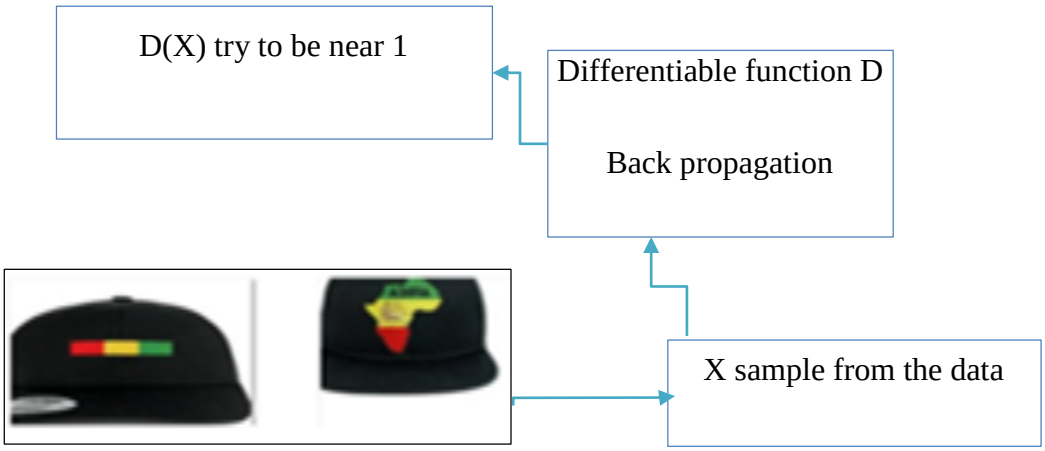


Figure 4.4 The role of a discriminator during generative network

Table 4.3 Algorithm for the generator network

1. Set the first sample to random Gaussian noise.
2. The generator produces output based on the initial form random Gaussian noise
3. Then discriminator classifies as “Real" and "Fake" as an output.
4. Calculate loss from discriminator classification at first.
5. while discriminator loss penalize if misclassify happens(until d classify incorrectly)

5A.Backpropagate discriminator then generator continues to get gradients.

5B. Use gradients by updating generator weights.

end

end

The following Figure 4.5 elaborates the discriminator network backpropagate to classify the real image as one (1) and fake images generated from the generator.

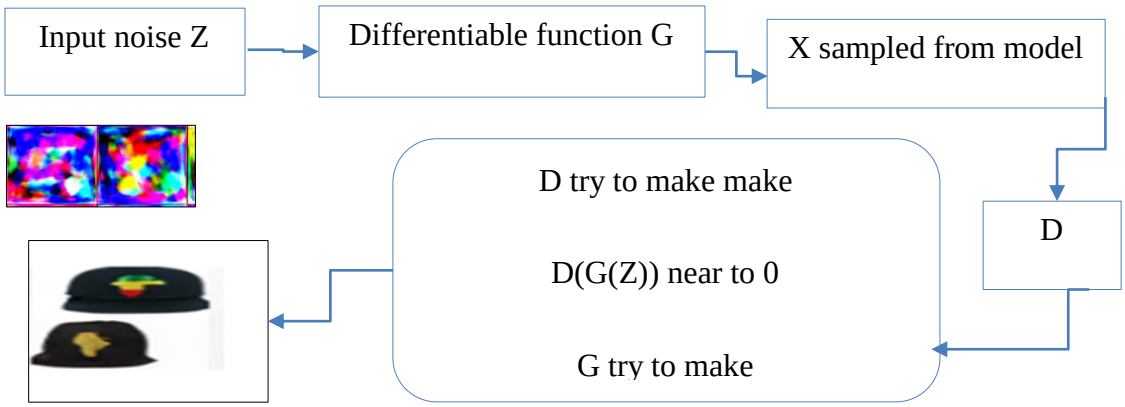


Figure 4.5 The role of a generator during generative adversarial training

This study used a progressive growing generative adversarial network for the stability of training. It extended ProGAN by adding multiple conditions such as mask, average color, and dimensional texture as an input for the generator. Such input of the generator helps to control a generated fashion image by tune texture, shape, and color. The detailed proposed architecture is presented in the following section.

4.3.2. Discriminator Training

The discriminator network classifies the real which is originally comes from a dataset labeled as one and the fake which is generated is generated labeled as zero throughout discriminator training [16] [19]. Table 4.4 shows the discriminator network disregards the generator loss, but it used discriminator loss only. The general task of the generator can be seen in three ways the first one classifying among real and fake.

Table 4.4 Algorithm for the discriminator network

1. The discriminator categorizes the real data as one and fake as zero.
2. while discriminator loss penalize if misclassify happens(until D classify incorrect way)

   The discriminator loss penalizes the discriminator for misclassifying a real as fake and vice versa.

The discriminator updates its weights through backpropagation from the discriminator loss through the discriminator network

end

end

Figure 4.4 elaborates on the discriminator network backpropagation during to classify the real image as one (1) and fake images generated from the generator.

4.4. Proposed Method

To generate a new fashion image that had a different fashion style generator using a generative adversarial network. This proposed solution improves the low-resolution results of previous literature.Secondly, to control the output by adding the class-conditions for generative adversarial network features. Fashion design focused on three attributes, namely shape, texture, and color. In Figure 4.7 below, shows the generator collectively used attributes such as texture, color, and shape as input for generated samples. The discriminator compared and classify as a real or fake based result generated from this network with a real image from the real dataset.

The generator of the progressive growing generative adversarial network starts from a Gaussian noise latent vector as it is shown in Figure 2.10. But the proposed architecture adds three conditions as inputs for the generator. The generator doing the same activity with proGAN but it considers shape, texture, and color whenever the new images are generated. The discriminator also the same activity that the discriminator of ProGAN does additionally to estimate the color generated by the generator and the original average color of the real images from the training set.

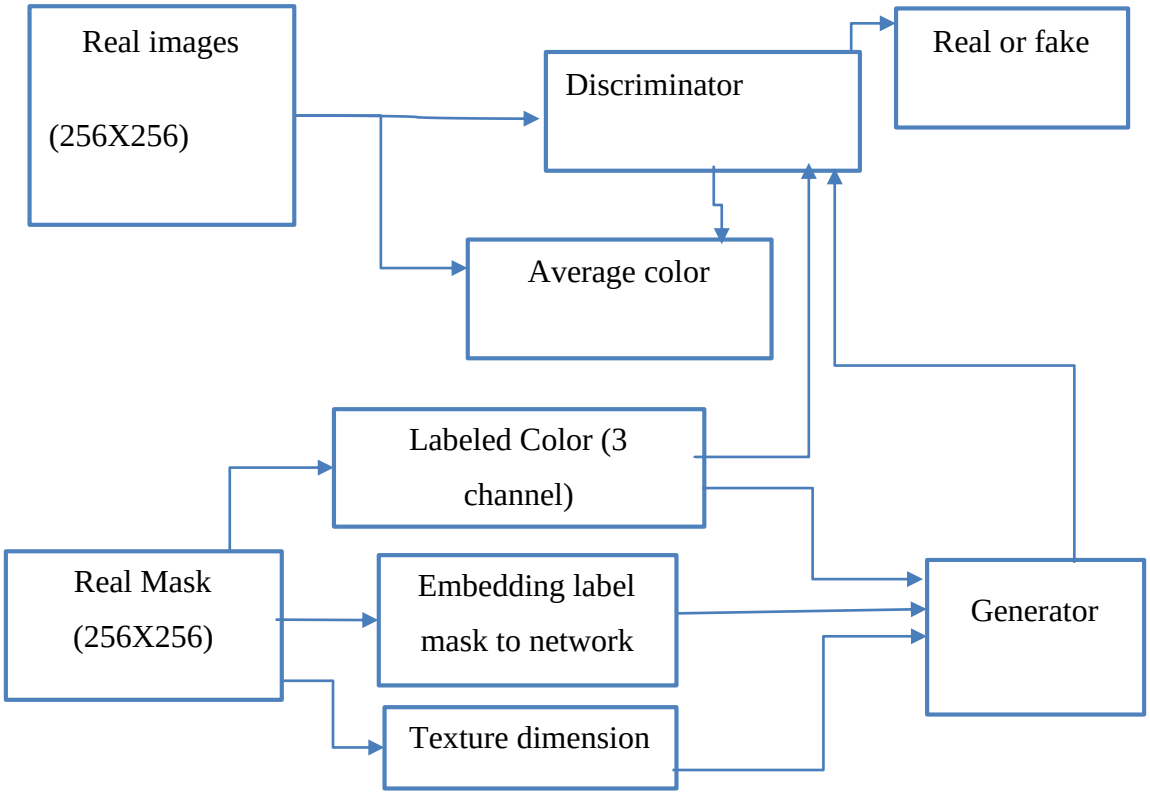


Figure 4.6 Proposed solution architecture during the training phase

4.4.1. Proposed Discriminator Architecture

The discriminator of our network almost the same with progressive growing GAN; likewise, a common function is not limited to classifying the real and fake or generated images[50], but also it distinguishes the average color and color generated by the generator (in 3 dimensions) as an auxiliary classifier. To train the proposed architecture used a Wasserstein generative adversarial network loss [21]. The loss of Wasserstein (Lw) Equation (4.1) used for loss of discriminator [2] that consists of the distance between the real distributed Pr(x) ~ $(E_{x\sim Pr}\lfloor D(x)\rfloor$ and generated or fake images Pr (x') ~ $(E_{x\sim Pg}\lfloor D(x')\rfloor)$.

$$L_w = E_{x\sim Pr}\left[D(\tilde{x})\right] - E_{x\sim Pg}\left[D(\tilde{x})\right] \qquad (4.1)$$

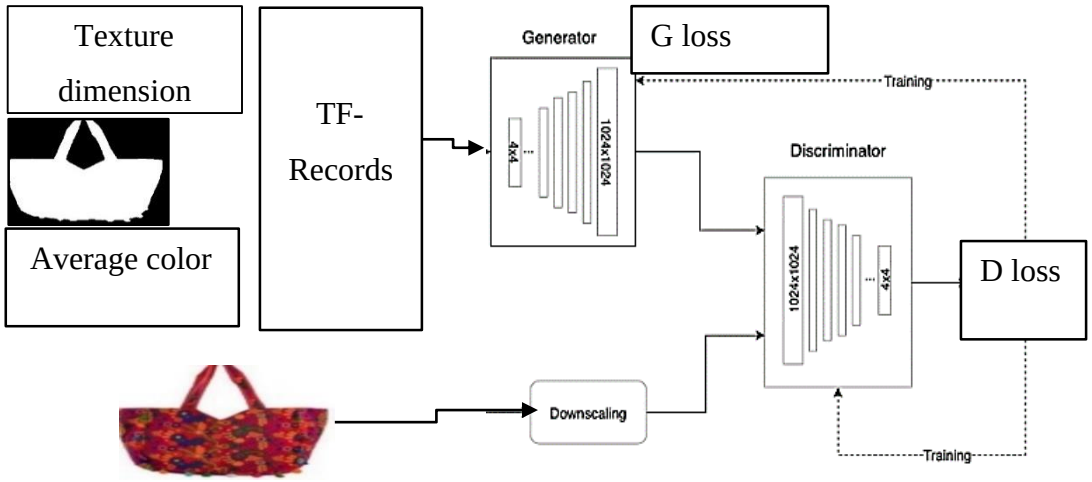
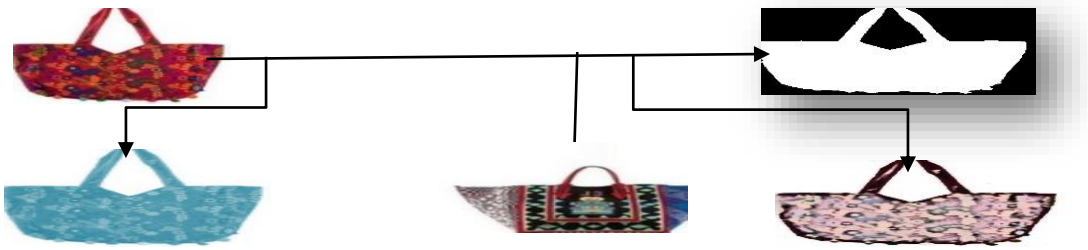


Figure 4.7 Proposed architecture for adding multiple conditions with ProGAN

4.4.2. Proposed Generator Architecture

The generator consists of such main attributes feed to ProGAN architecture as stated in Figure 4.7. The first attribute is the average color of an image c function p (c) consists of a 3D-dimensional vector of color images. The p(c) normalizes to a uniform arrangement in the interval between [-1, 1] and merged with color information written into a pickle file to generate a color.

The second attribute given to the generator network is the texture which is a 512- a dimensional latent vector consists of the local pattern structure t function p (t) in which a normalized mean and variance. The last input is the binary segmentation mask to generate customer shapes. In Figure 4.8 below, three inputs merged to high dimensional vectors that through passed through the generator network $\check{Z} = G(\text{Color}, \text{texture}, \text{shape})$ where $\check{Z}$ generator .



a) Color change [average color] b) texture [texture dimnision] c) shape[mask]

Figure 4.8 Image synthesis control of shape, color, and texture

4.4.3. Color Synthesis Control

The above Figure 4.8 (a) generating an image with the same texture and shape. The following Equation (4.2) show the average color computation of a real or fake article [11] using its corresponding segmentation mask where |s| summation of segmented mask s(i,j),x(i,j) computed real image pixel at a specific (i,j) location Source Adapted from

$$A(x.s) = \frac{1}{|s|}\sum_{i,j}^{n} s(i,j)x(i,j) \qquad (4.2)$$

This average calculation can also mater on color loss calculation as it is shown in the following Equation(4.3) suppose two images with constant color and remains unchanged ${x_1}^{-c} = G1(c,t,s)$ and ${x_2}^{-c} = G2(c,t2,s2)$.

$$L_c = E_{c\sim p(c)}\left[E_{{x_1}^{-c},{x_2}^{-c},{p_g}^{-c1},}\left[A\left({x_1}^{-c},s1\right)\left({x_1}^{-c},s2\right)\right]\right] \qquad (4.3)$$

4.4.4. Shape Synthesis Control

The above Figure 4.8 (b) keeping shape consistency by generating pixel location background$x^{-} = G(c,t,s)$. The following Equation (4.4) illustrates the loss added to the generator network to keep the consistency of the shape during the synthesis.

$$L_s = E_{x\sim p_g, s\sim p_s,}\left[\frac{1}{1-s}\sum_{i,j}1-s|x'-b|\right] \qquad (4.4)$$

Where E_x for total real data distribution , P_g for generated data ,1-s for the binary segments mask complement of the input segmentation, and b for the background color, which is white.

4.4.5. Texture Synthesis Control

The above Figure 4.8 (c) generating an image with the same texture and shape. Before computing the loss of texture and also compute V (), the flattens an input image into a 128^2 ×3 channel matrix, and $S(\cdot)$ computes the Laplacian matting matrix (1282×1282). The following Equation (4.5) and (4.6) illustrated the loss added to the generator network to reduce the texture inconsistency problem. It used the following loss function:

$$v_1^{-t} = v\left(x_1^{-t}\right), s_1^{-t} = s\left(x_1^{-t}\right) \tag{4.5}$$

$$L_t = E_{t\sim p(t)}\left[E_{x_1^{-t}, x_2^{-t}, p_g^{-t},}\left[Tr\left(v_1^{-t^T} s_2^{-t} v_1^{-t} + v_2^{-t^T} s_1^{-t} v_{12}^{-t}\right)\right]\right] \tag{4.6}$$

Where L_t is the consistency loss of texture, and Tr (·) is the trace of a matrix of local structure

4.4.6. Generator Average Color Check

To avoid the collapse of the average colors into a single particular color it needs to add additional loss that can monitor such issues in the training process. In Equation (4.7) shown below to the stability mechanism check to a generator by adding color.

$$L_{gcolor} = E_{c\sim p(c)}\left[E_{c^{-}\sim p^{c}{}_{g}}\left[\left\|c - A\left(x_1^{-c}, s1\right)\right\|\right]\right] \tag{4.7}$$

Where L_{gcolor} the generator check color for preventing from diminsing to a single colors see the effect of this functions matters in the results. The E_c the real data distribution color aproximaly with the generated one p_c but p_c^g color information for generated datasets.

The overall generator loss In Equation (4.8) below, consists of various inputs weight, the sum of Wasserstein loss, and the auxiliary color difference of real and fake generated through a segmented mask during the training. The summation of training parameters a summary of each loss as shown as follows.

$$L_G = \min_G L_w + L_{aux} + \lambda_c L_c + \lambda_s L_s + \lambda_t L_t + \lambda_g L_{gcolor} \tag{4.8}$$

Where L_G the overall loss of generator, L_w is Loss of Wasserstein, L_{aux} the auxiliary color difference between real and fake, λ_c is weight weights of color, λ_t is a weight of texture and λ_s is a weight of shape.

4.5. Style-based Generative Adversarial Network

This fashion style generation method is based on artistic style transfer works. This research is mainly based on Style based generative adversarial network. A styleGAN new generator is an extension of progressive generative adversarial (ProGAN) that improves the state-of-

the-art in case of traditional data distribution quality metrics [21]. In mapping network, random vector (z) to the generator is projected for the middle latent space (w) as shown in Figure 4.8 below for feeding network mapping [22] become this method could be considered as collective style transfer and generative adversarial network by adding adaptive instance normalization.

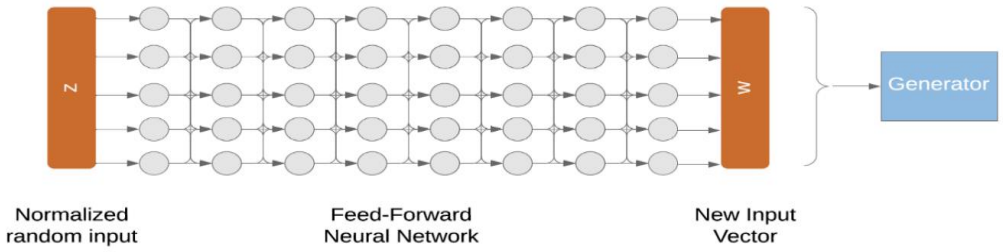


Figure 4.9 Adaptive instance normalization of the style w

Even though styleGAN could generate high-resolution and quality images but control with a certain condition not addressed and also the architecture is too complex to do so. The following Figure 4.9 demonstrated the detail architectural difference between the progressive growing GAN and Style based generative adversarial network. Since both of this paper addressed by the same first author and same institution NVIDIA.

The resolution of the synthesis starts from 4 × 4 progressively growing architecture up to 1024 × 1024 for the stability of generative adversarial network training. Style GAN can further improve the image synthesis high-quality of StyleGAN through the redesigned generator and loss function that measures deviates from the training set [21] as described in Figure 4.10.

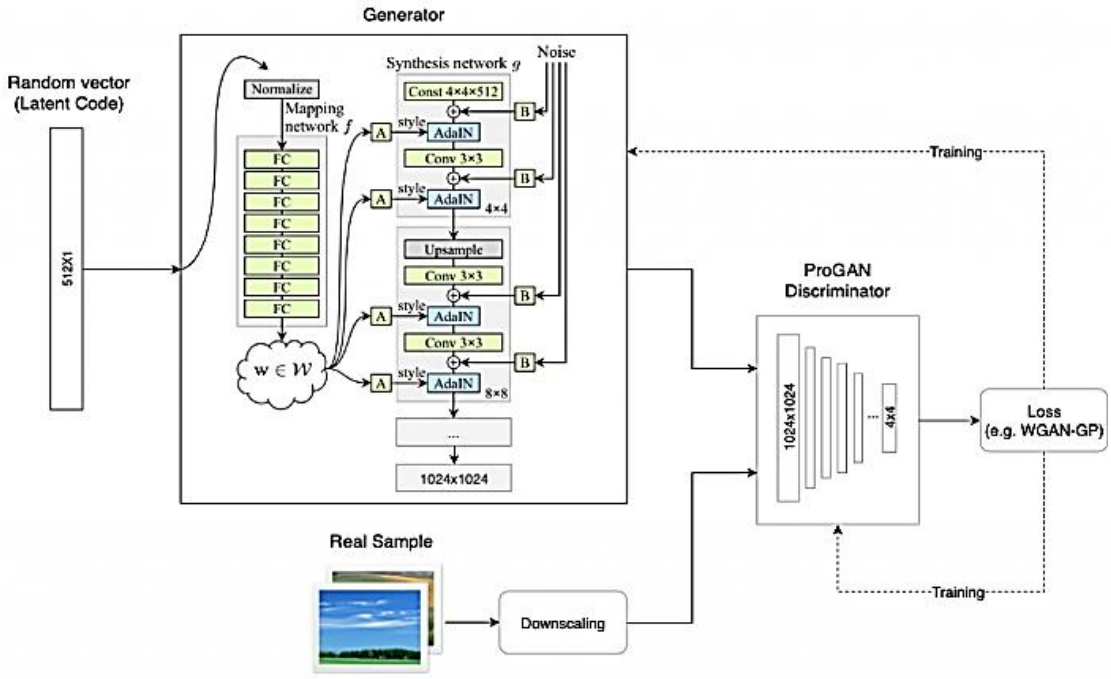


Figure 4.10 Architecture discriminator and generator Style based GAN architecture

4.6. Conditional Style-based Generative Adversarial Network

Even though such class revised were done the style-based generative adversarial network, it still is difficult to control using some conditional settings [57]. For this reason, this proposed extend style GAN by adding a single class condition to the network to create an image with a specific class or label with the same architecture and collectively given as an input for style GAN architecture shown as in Figure 4.10 below.

The ultimate difference between the proposed method and StyleGAN architecture is how the information to the generator w created, and the discriminator calculates the loss based on additional conditions label. A class condition merges the input z and mapping network. In the above figure, the drawback of the style based generative adversarial network is difficult to control conditional features. This proposed solution added class conditions to improve the low-resolution results of previous literature in fashion GAN by producing high-quality images and, secondly, to increase the controllability of the style GAN conditional nature of the generative adversarial network.

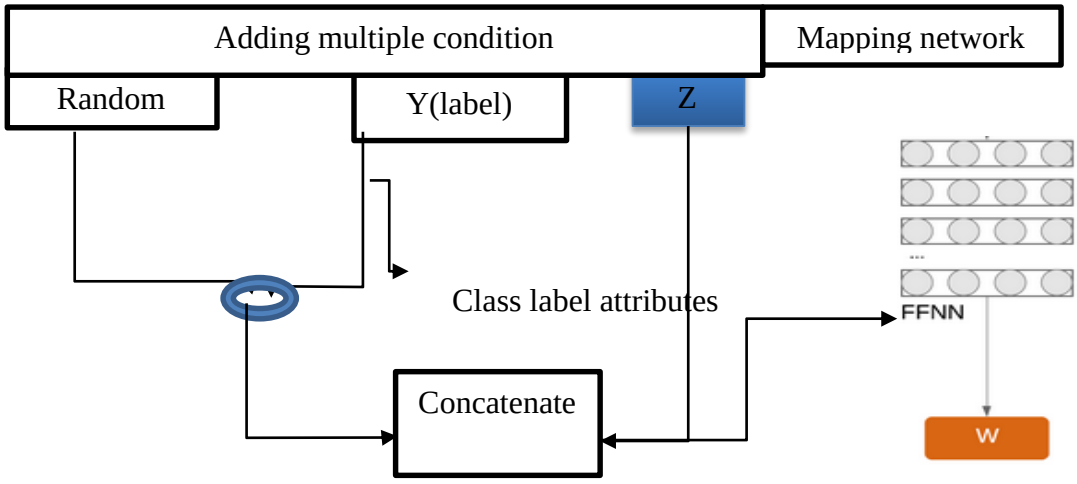


Figure 4.11 Proposed method architecture with specific class condition

CHAPTER FIVE

5. IMPLEMENTATION DETAILS

5.1. Overview of the Implementation

The goal of the proposed solution was achieved after the implementation of tasks. The prepossessing, augmentation, label, and annotation class conditions explained first, then the training implementation follows. This chapter describes the data augmentation, color computation, and labeling, mask generation steps along with code segments. In this section, important segments of code were presented whereas the important detail codes illustrated in the appendix section.

The second could be applying a personal dataset on the existing original style-based generative adversarial network. The last from data augmentation will discuss in this chapter which tasks implemented the personal dataset by adding a bit condition on style GAN (conditional style GAN). Last, this chapter discussed using a pre-trained model, the evaluation metrics implementation with cross ponding code sections.

5.2. Working Environments

5.2.1. Desktop Computer with Hardware Utilities

- Hard-Disk: configured terabyte added two tetra byte of storage.
- SSD: configured SSD is 500GB.
- Processor: Intel(R) Core i5 dell-3090 CPU, 4 Core(s), 4 Logical Processor(s)
- Graphical purpose processing Unit: for high-speed time computation during training. 2 RTX 2070 super installed in separated desktop.
- Installed Memory: Physical Memory (RAM) 8.00 GB added 6, totally configured 14.00 RAM.

5.2.2. Desktop Computer with Software Configuration

- OS: Microsoft Windows 10 Pro, x64-based PC, Version 10.0.17134 Build 17134
- System Model: - Dell OptiPlex 3090.
- GPU optimizer: the latest version of the NVIDIA GPU optimizer and scheduler.
- CUDA toolkit

- CuDNN

In this thesis, RTX 2070 Super 8GB dedicated graphics and 12 GB RAM used to fasten training.

Figure 5.1 GPU configuration as components in other systems

5.3. Setup Environments

5.3.1. Application Software

Anaconda: In an application used to install the latest version of python with its different module and IDEs, for implementing proposed solution an anaconda application version 1.5 with 64-bit support used.

5.3.2. Integrated Development Environments and Editors

Jupyter Notebook: - is an IDE that is the most popular and recommended IDE to researchers to edit python code, to visualize some diagram, training, and testing charts and processing progress. To implement the proposed solution, the Jupiter notebook with a 6.0.0 version is used, which is installed through anaconda installation.

Colab: -A cloud-based Jupiter notebook can enhance computing time using the Tensor Processing Unit (TPU), General processing Unit (GPU), Random Access Memory, and Large storage space to upload data set for training the results until 9 hours' time session.

Visual-Studio-Code: - is an IDE that is easily configured with different python environments and used to edit python code, c++ code, and another programming language

code also editable using this IDE. A version 1.38.1 with the community and 64-bit support installed for implementation in addition to the Jupiter notebook.

5.3.3. Programming Language and Module Libraries

- Python: - to implement the proposed solution python programming language with version 3.5 used due to the version incomparability. Many deep learning modules does not install properly on the latest version of python.
- Keras with version 2.2.4, which used to load the pre-trained model, used to apply convolution, pooling, and other deep learning processes.
- Tensorflow with version 1.15.0 and above used to connect the retrained model and feature fusion model graphs Tensor board with version 1.15.0 and above to visualize training progress in deep learning, in the proposed approach to visualize the feature fusion progress.

5.4. Training Procedure

The training procedure in this proposed research

- Collect massive dataset from a different source
- Pre-processing dataset
- Augmenting in a different angle of point and scale.
- Preparing the dataset (label the image in excel) and write into a pickled dictionary.
- Create TF-records, which is a machine-readable format.
- Configure the GPU and dataset configuration along with the label pickle file.

Let see all the above set of training procedures in a step-by-step and detailed manner.

5.5. Implementation Techniques for Dataset Preprocessing

5.5.1. Dataset Description

The original dataset was too small and with different sizes and backgrounds. Since then, the image augmentation is used to generate images with different backgrounds, sizes, angles, using augmentation techniques to cover some possible scenarios that make the generation process easy. Figure 5.1show the datasets collected from a different source in a class or category.

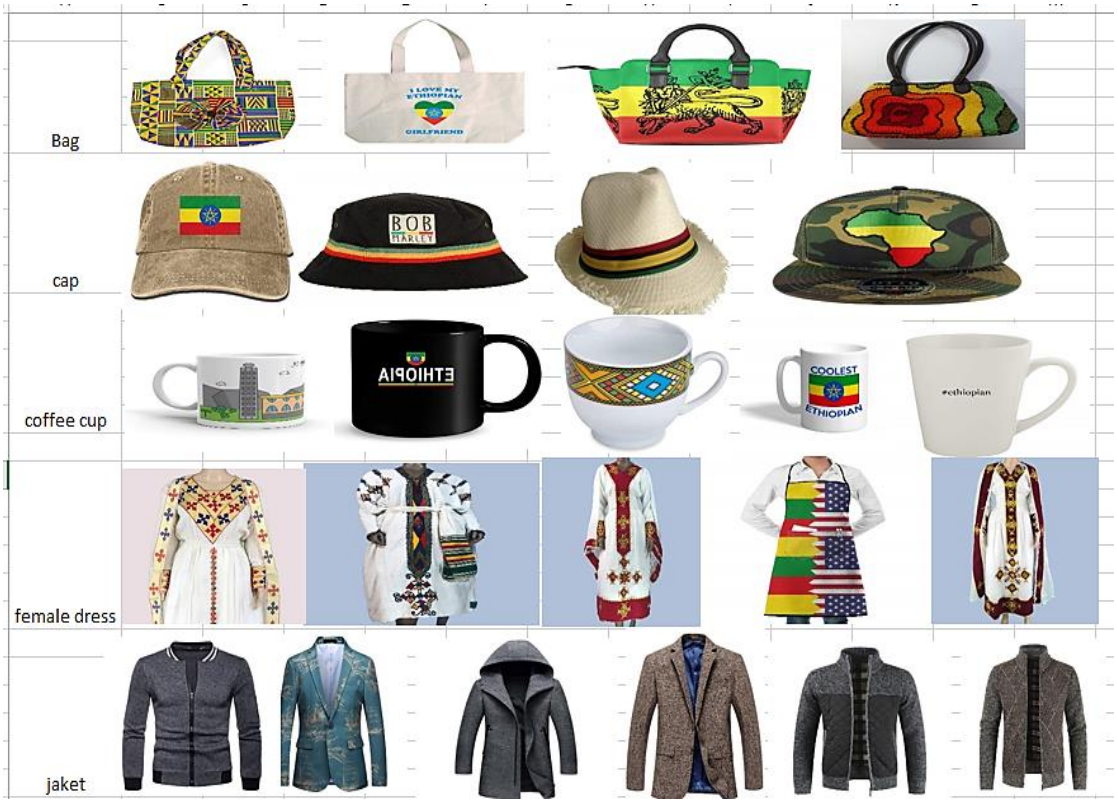


Figure 5.2 Sample image dataset

5.5.2. Preparing the Dataset

The dataset e converted to Tensorflow record files (TF_records). The primary task to do so, each image in the data set must have the same format in terms of size, extension, color space, and bit depth. Any unequal images will discard automatically from the dataset.

```
data = pd.read_csv (r'C:\Users\given\Desktop\code\label.csv')

df = pd.DataFrame(data, columns= ['Filenames','Labels'])

pickle_off = open("average_colors.pkl", 'rb')

label = pickle.load(pickle_off)

print(label)
```

Sample code 5.1 Labeling dataset into pickle dictionary

The above code segment takes an excel file containing a file name and corresponding class label number of the dataset to write to a pickled dictionary for machine readability.

The image augmentation task is quite mandatory due to deep learning or generative adversarial network needs for the efficiency of results. The common augmentation operations are rotation and cropping.

```
datagen=ImageDataGenerator(
rotation_range=0,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')

here is a condition to kicked out overcropped images
```

Sample code 5.2 Data augmentation operations

5.5.3. Implementation of Color Computation and Labeling

The proposed method used an average color as an extra input condition for the generator network. The following Equation (5.1) and (5,2) show the normalizing in the range in [-1, 1] and [0, 1], the difference between the actual image color value and maximum divided by the difference between the maximum 255 and the minimum 0.

$$normalize[0,1]\frac{x-xmin}{xmax-xmin} \rightarrow x/255 \qquad (5.1)$$

$$normalize[-1,1] = 2*(\frac{x-xmin}{xmax-xmin})-1 \rightarrow 2(\frac{x}{255})-1 \qquad (5.2)$$

The following figure show code snip for computing the average color of the dataset. This average computation used for a color channel databased that can be used during the generation process.

```
Read_image = cv2.imread(file)/(255)

Read_image           =           2*(cv2.imread(file)/(255))-1
avg_color_per_row       =       np.average(gray,       axis=0)
average
=np.average(avg_color_per_row,axis=0).astype('float32')
 bank_dict.update({x[1]:dominant})
```

Sample code 5.3 Average color computation and labeling

5.5.1. Binary Mask Segmentation

Creating a binary mask segmentation by setting a threshold value to 0 using OpenCV utility packages for the background pixels and 255 for exact image shapes. This summing up the threshold and binary inversion better mask.

```
path = 'orginal'

outPath = 'masks'
files = glob.glob(path + '/*.jpg')
for file in files:
    print(file)
    x=file.split('\\')
    img = cv.imread(file)
    gray = cv.cvtColor(img,cv.COLOR_BGR2GRAY)
    ret, thresh =
cv.threshold(gray,0,255,cv.THRESH_BINARY_INV+cv.THRESH_OTSU)
    cv.imwrite(outPath + '/'+x[1], thresh)
```

Sample code 5.4 Binary mask segmentation

5.5.2. Create TF-Records

To add multiple conditions by using labels that can refer to the corresponding file name and class label using excel for the sake of writing to pickle dictionary. Since then, computing the average color of the dataset and binary mask generated. The TF_Record is created from these collected conditional inputs namely original images, average color, and binary mask. The following sample code shows how the color, shape, and mask are created into single TF-records, whereas in progressive growing GAN the TF-Record is created from original images.

```
 def add_image_mask_color(self, image, mask, color):

 ex    =tf.train.Example(features=tf.train.Features(feature={

'image':tf.train.Feature(bytes_list=tf.train.BytesList(value
=[image.tostring()])),

'mask':tf.train.Feature(bytes_list=tf.train.BytesList(value=
[mask.tostring()])),

'color':tf.train.Feature(bytes_list=tf.train.BytesList(value
=[color.tostring()])),
            'image_shape':
tf.train.Feature(int64_list=tf.train.Int64List(value=image.s
hape)),
            'mask_shape':
tf.train.Feature(int64_list=tf.train.Int64List(value=mask.sh
ape)),
            'color_shape':
tf.train.Feature(int64_list=tf.train.Int64List(value=color.s
hape)), }))

self.tfr_writer.write(ex.serializeToString()
```

```
for i, image_path in tqdm(enumerate(image_filenames), total=len(image_filenames)):

        basename = os.path.basename(image_path)

        mask_path = os.path.join(mask_dir, basename)

        image = read_image(image_path, dataset_resolution, PIL.Image.ANTIALIAS)

        image = read_image(image_path, dataset_resolution, PIL.Image.ANTIALIAS)

        mask = read_image(mask_path, dataset_resolution, PIL.Image.NEAREST)

        color = labels[basename]

        tfr.add_image_mask_color(image, mask, color)
```

Sample code 5.5 Organize the conditional inputs before TF-Record created

The script is run from the terminal to create the machine-readable file that stores a sequence of binary records through takes the paths of images and directory as flags.

5.6. Image Generation Implementation

5.6.1. Training Configuration

```
desc += '-ethio_fashion; dataset =
EasyDict(tfrecord_dir= ethio_fashion, resolution=512);
train.mirror_augment = False;# Dataset configuration

submit_config.num_gpus = 1;# Gpu configuration
sched.minibatch_base = 4; sched.minibatch_dict = {4: 128, 8:
128, 16: 128, 32: 64, 64: 32, 128: 16, 256: 8, 512: 4};

sched.D_lrate_dict = EasyDict(sched.G_lrate_dict);
train.total_kimg = 90931

desc += '-nogrowing'; sched.lod_initial_resolution = 128;
G_loss.weights.color_consistency = 100.0
G_loss.weights.texture_consistency = 100.0
G_loss.weights.shape_consistency = 1.0
G_loss.weights.generator_color_check = 100.0
We call the above weights see the detail here
```

5.6.2. Training Hyperparameter

The training parameters specify the images, the number of epochs, and batch size along with the number of GPU ,dataset,ephoch,batch size configuration were mentioned as follows;

```
def train_mutiple_condtional_progressive_gan(
    G_smoothing = 0.999,
    D_repeats = 1,
    minibatch_repeats = 4,
    mirror_augment = False,
    drange_net = [-1,1],
    image_snapshot_ticks_ephoch = 1,
    network_snapshot_ticks = 10,
    save_tf_graph = False,
    save_weight_histograms = False,
```

Sample code 5.6 Setting train hyperparameter

5.6.3. Training Steps

The training iteration writes to the pickle file and saves the updates through time in the following summarized manner.

```
print('tick %-5d kimg %-8.1f lod %-5.2f minibatch %-4d time %-12s sec/tick %-7.1f sec/kimg %-7.2f maintenance %.1f' % (
 tfutil.autosummary('Progress/tick',cur_tick),
tfutil.autosummary('Progress/kimg', cur_nimg / 1000.0),
tfutil.autosummary('Progress/lod', sched.lod),
tfutil.autosummary('Progress/minibatch', sched.minibatch),
 tfutil.autosummary('Timing/total_days', total_time / (24.0 * 60.0 * 60.0))
 tfutil.save_summaries(summary_log, cur_nimg)
```

Sample code 5.7 Displaying training progress reports

5.6.4. Network Snapshot and Checkpoint

In each training, the result has two outputs. The first one is the generated images and learned network saved in each iteration.

```
gw = 1; gh = 1
    if size == '1080p':
        gw = np.clip(1920 // G.output_shape[3], 3, 32)
        gh = np.clip(1080 // G.output_shape[2], 2, 32)
    if size == '4k':
        gw = np.clip(3840 // G.output_shape[3], 7, 32)
        gh = np.clip(2160 // G.output_shape[2], 4, 32)
```

Sample code 5.8 Save to 4k images grid

```
if cur_tick % network_snapshot_ticks == 0 or done:
  misc.save_pkl((G, D, Gs), os.path.join(result_subdir, 'network-snapshot-%06d.pkl' % (cur_nimg // 1000)))
```

Sample code 5.9 Saving the network model in each 10 epoch

5.6.5. Pre-trained Model Implementation

Once the model is trained for generating an image using a pre-trained model is quite an easy task. Not only to generate images from the model but the model also used to resumed whenever the power is gone. The pre-trained is easy to use, and flexible compatible and also requires less computational resources like training. The following code segment is used to generate images based on the learned network given by the model. Suppose if we want to generate 5000 images from the trained model.

```
tflib.init_tf()
    _G, _D, Gs = pickle.load(open("results/model.pkl", "rb"))
     Gs.print_layers()
     for i in range(0,5000):
         rnd = np.random.RandomState(None)
         latents = rnd.randn(1, Gs.input_shape[1])
         fmt = dict(func=tflib.convert_images_to_uint8,
nchw_to_nhwc=True)
         images = Gs.run(latents, None, truncation_psi=0.6,
randomize_noise=True, output_transform=fmt)
         os.makedirs(config.result_dir, exist_ok=True)
         png_filename = os.path.join(config.result_dir +
'/finished', '1' + str(i) + '.png')
```

5.7. Implementation of Model Evaluation

5.7.1. Frechet Inception Distance (FID50k)

The Frechet inception distance calculated by computing the difference between the real image and generated images

```
# Calculate FID. Difference between the real and fakes
         m = np.square(mu_fake - mu_real).sum()
         s, _ = scipy.linalg.sqrtm(np.dot(sigma_fake,
sigma_real), disp=False) # pylint: disable=no-member
         dist = m + np.trace(sigma_fake + sigma_real - 2*s)
         self._report_result(np.real(dist))
```

5.7.2. Perceptual Path Length (PPL90K)

The perceptual path length is calculated by computed the difference between two consecutive images at interpolating latent inputs.

```
img_e0, img_e1 = images[0::2], images[1::2]

distance_measure = misc.load_pkl('vgg16_zhang_perceptual.pkl')
#
distance_expr.append(distance_measure.get_output_for(img_e0,
img_e1) * (1 / self.epsilon**2))
```

CHAPTER SIX

6. RESULT, EVALUATION, AND DISCUSSION

6.1. Introduction

This chapter presents some of the results of the implementation process such as data augmentation, mask segmentation, and main training results on generative adversarial network algorithms on the same fashion image datasets. It also discusses the output generated by the proposed algorithm and comparing with others, namely Style GAN, conditional Style GAN through different evaluation metrics such as inception distance (FID) and path length (PPL). Moreover, it also discusses the human evaluation user study metrics along with both paired and unpaired study.

6.2. Results of the Study

6.2.1. Dataset Preparation Result

The original dataset collected was too small and an image augmentation used for increasing the dataset size different angle points for single images. Table 3.2 presentes the dataset size after augmentation is 7.5 times as of the original dataset. The data agumentation results here bellow only show rotation with 0 dgree meanshift to the left , right , and shear range with the values 0.2 .

Figure 6.1 Dataset agumentation results

6.2.1. Results of the Data Augmentations

The data augmentation could enhance the training performance of the varieties augmentation in single image data. This could be easy for the training of the generative model to learn easily from the varieties of the data distribution. The data agumentation results here bellow only show rotation with 40 dgree meanshift to the left , right , and shear range with the values 0.2.

Figure 6.2 Augmented image samples:

where a) is the original image, b) is a rotated image of the original image at 40◦, c) original width shift to the right in 0.2, d) original width shift to the left, e) is flipped image that is horizontally.

6.2.1. Result in Mask Segmentation

This study used a mask segmentation result as in the input of the proposed generator. Binary mask segmentation results obtained from summing the out-threshold and binarized image. It results used to control the shape synthesis in the generation process. Figure 6.4 demonstrates the result of the binary mask for sample images.

Figure 6.3 Result of mask segmentation for corresponding images

6.3. Experimental Results

Three experiments were performed in the same hardware and software, hyperparameters, and dataset configuration. The first and the last experiment proposed solutions by adding an extra condition in both progressive growing generative adversarial network and style based generative adversarial network. But the second the existing style based generative adversarial network with any modification by adjusting the same training parametric configuration.

6.3.1. Conditional Progressive Growing Generative Adversarial network

The following images are not original and don't exist instead created by the conditional progressive generative adversarial network.

Figure 6.4 Fashion images generated from conditional ProGAN

6.3.2. Conditioning Results

This multi-conditional progressive growing had limited control to shape from a binary mask, color from average color, and texture from 512 dimensions.

Table 6.1 Multiple condition fashion images generations

SN	Original	Generated	Condition by texture G(t)	Condition color and shape G(c,s)
1				
2				
3				
4				

Table 6.1 shows that multiple conditions are given to the generator and produce images along with corresponding on the condition. The conditional inputs were successfully able to generate a diverse shape, texture, and color many repetitions with a small difference.The diversity datase inbalancement a cross the class had significant effect on the quality of generated images. To examine this the researcher done a single class data experments and achived remarkable results.

a)

A) Generated images

b)

B) Orginal images

Figure 6.5 Comparison between proposed method's generated(aA) and real images(B)

6.3.3. Conditional Style based Generative Adversarial Network

The following results were generated from the style-based generative adversarial network by adding a class condition.

Figure 6.6 Generated output images with different shape, texture, and color

6.3.1. Failed Results

In all three cases, multiple failed result was achieved. These bad examples occurred due to various reasons. The first would be irregular shapes and unidentified shapes along with a different class in the dataset distribution. But the second could be the appropriate color average conditions.

Figure 6.7 Failed generated results due to irregular shapes, texture, and invalid color

6.4. Evaluation Metrics

Three quantitative evaluation metrics used to evaluate the generated images and model.

6.4.1. Frechet Inception Distance

Table 6.2 shows the evaluation results of Frechet inception distance of the learned model saved at the last session during training for the three experments. This distance is calculated for randomly selected 50,000 images from the whole training sets. The lower FID shows the image became close more realistic where the more FID the image is the less realistic and huge gap between the real and generated images.

Table 6.2 Frechet inception distances comparison between for all experiments

S.No	***Experiments***	***FID50K***	***Time***	***the same Dataset***
1	Condition PGGAN	41	14m 7s	Yes
2	StyleGAN(Exp2)	46.3	12 m 0s	Yes
3	Condition StyleGAN	58.9	31m50s	Yes

6.4.2. Perceptual path Length

Table 6.3 shows the comparison results of perceptual path length (PPL) and its corresponding time taken to evaluate across three experiments in the same dataset.

Table 6.3 Comparison between perceptual path lengths for all experiments

S.No	***Experiments***	***PPL90k***	***Time taken***	***The same Dataset***
1	Condition PGGAN	1500	25 m 27 sec	Yes
2	StyleGAN(Exp2)	1966.52	31m 21 sec	Yes
3	Condition StyleGAN	2234.6	24 m 55 sec	Yes

The detail perceptual distance comparison is presented in the appendix section of PPL

6.4.3. Human Evaluations Results for Paired User study

The interested user fills Google questionaries' by given two paired data as to label as fake and real as to label as fake and real based criterion texture shape and color consistency.

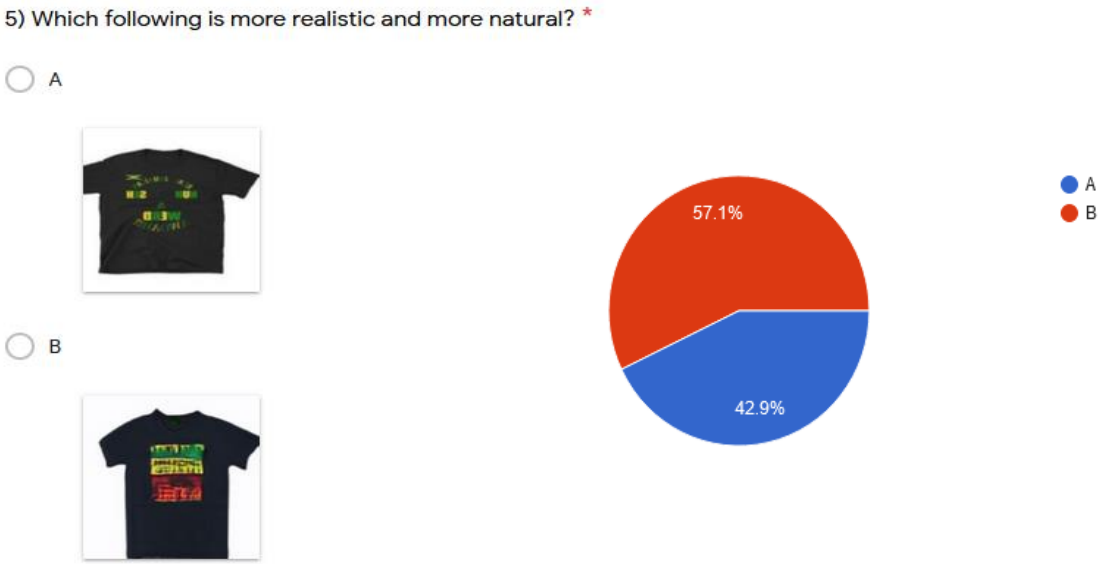


Figure 6.8 Human evaluation for a paired user study

Table 6.4 Summary of a human evaluation user study in pared data settings

	Summary of Evaluation					
Actual	B	1	A	B	A	
User correct guess	10	7	4	9	5	35/13
User correct guess / total user	10/13	7/13	4/13	9/13	5/13	(2.69)/5=0.538
User confused	3	6	9	4	8	(30/13)
User confused/ total user	3/10	6/13	7/13	9/13	8/13	(2.307)/5 =0.462

The detail Human evaluation presented in the appendix section of a paired user study

6.4.4. Human Evaluations Results for Unpaired User Study

The interested user fills a Google questionaries' single data as to label as fake and real based on the criterion for the consistency of texture, shape, and color as shown as follows.

Figure 6.9 Human evaluation for an unpaired user study

While comparing with an unpaired evaluation with paired evaluation users had a greater chance to confuse user a single image as real or fake. As a result, the user didn't know the real and fake (confused) 48% and correctly guess 52%.

Table 6.5 Unpaired user study observed from Google form questionnaires.

Actual	A	A	A	A	B	***Sum / Total***
User correct guess	6	7	8	6	7	34/13=2.61
User correct guess / total user	6/13	7/13	8/13	6/13	7/13	2.61/5=0.524
User confused	7	6	5	7	6	31/13=2.38
User confused/ total user	7/13	5/13	5/13	7/13	6/13	2.38/5=0.476

The detail Human evaluation presented in the appendix section of an unpaired user study

6.4.5. Human Evaluation Quality and Assessments

This evaluation is based on user evaluation metrics such as texture, shape, and color. Table 6.6 shows the average of uses participates in Google form evaluations.

Table 6.6 Comparison of human quality assessments

Criterion fields	***ProGAN***	***StyleGAN***	***Conditional StyleGAN***
Shape consistency	3.33	2.77	2.88
Color consistency	2.88	2.67	2.77
Texture consistency	3.55	3	3
Beauties	2.66	2.88	2.66
Overall average	**3.11**	**2.83**	**3.02**

6.5. Discussion

6.5.1. Discussion on Frechet inception Distance Results

For the performance and quality evaluation purpose, a Frechet inception distance was performed. Figure 6.9 shows the evaluation of three experiments FID50K results where the

proposed solution score 41 at a time take to evaluate 14.7 minutes. This shows that conditional ProGAN results are statistically close as to real images as lower FID scores.

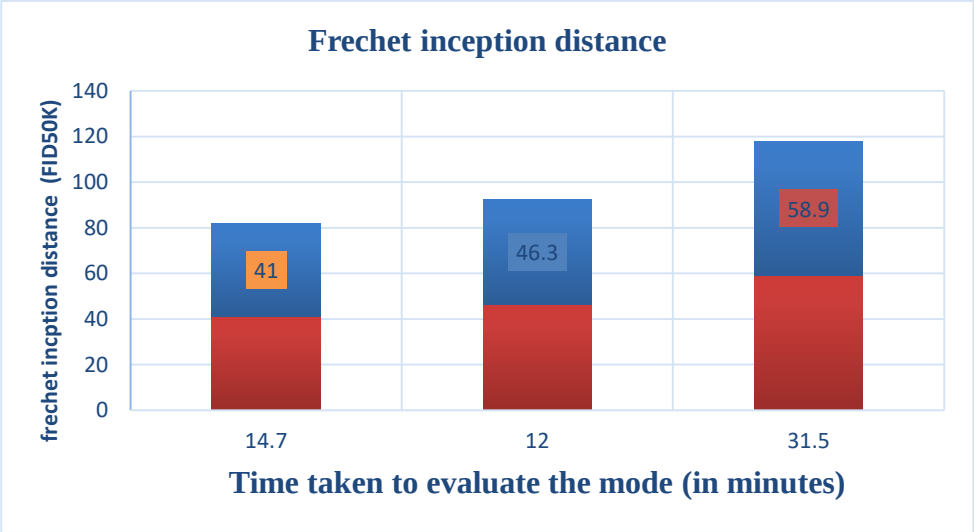


Figure 6.10 FID50k comparison of the three experiments model

6.5.2. Discussion on Perceptual Path Length Results

The result of PPL for conditional ProGAN shows the interpolation distance difference between two consecutive images is 1500. That means the VGG16 embedding of images close to each other. The higher PPL the more the two images are different from each other.

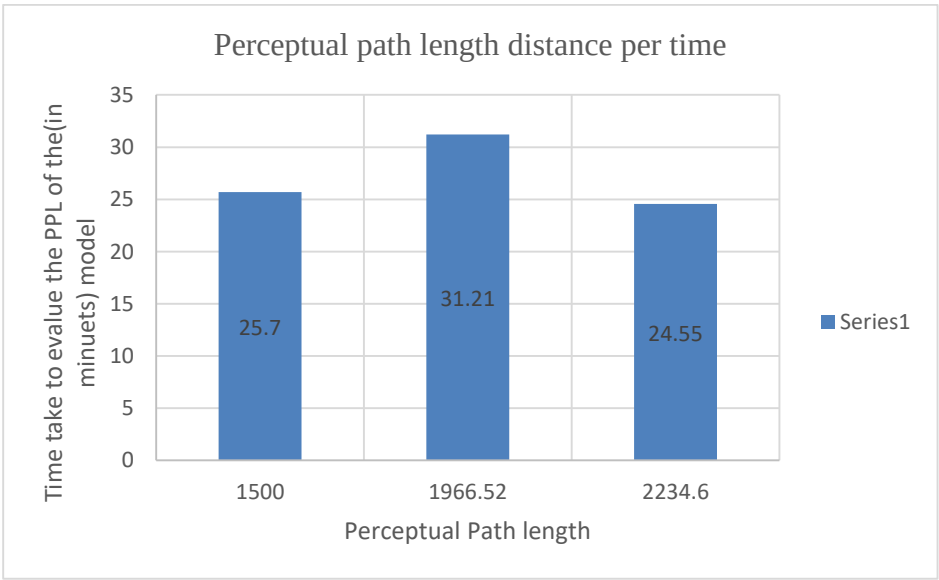


Figure 6.11 PPL90k comparison of the three experiments model

6.5.3. Discussion on Human Evaluation Results

In this evaluation, users had a chance to compare the more realistic images from two alternatives. Based on the user study evaluation some users label the real as fake and the fake as real. The average paired user study confusion rates rate of 46% and correctly guess 54%. Whereas the unpaired user studies confusion rates rate of 47.6% and correctly guess 52.4%. The result based line for the generative adversarial network is 50%.

6.5.4. Discussion on Human Evaluation Quality Assessments Result

The human quality evaluation assessment is the average values of individuals given from Scale [1, 5] scale based on texture, shape, color, and internal beauty of the generated images. Appendix

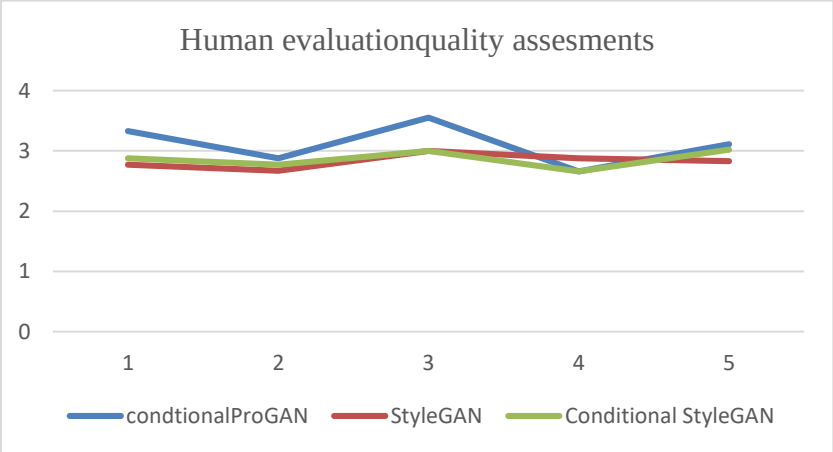


Figure 6.12 Human evaluation and quality assessments

CHAPTER SEVEN

7. CONCLUSION AND FUTURE WORKS

7.1. Conclusion

Currently, image generation has shown successive results in generating a realistic look of images from a real image dataset distribution. This can be applied in various domains such as fashion design. On the other hand, creating high-resolution images was also critically an issue. Besides this, fashion image generation tasks were prone to subjectivity and personalization of design in terms of shape, color, and texture. To overcome this problem integrating such low-level design attributes in the generation process could be an alternative solution. In this study, an architecture is developed to integrate the controlling inputs such as color, shape, and texture with the existing ProGAN architecture.

This study presents a generative adversarial based fashion design along with multiple conditional attributes. To accomplish this, an average color of all the dataset is computed and pickled into a dictionary; a binary mask is generated from threshold values to keep preserve the shape organized TF-Records. In addition, a 512 local structure of the image was taken to show the texture dimension of the images. These three conditional attributes are integrated with the existing generator architecture of progressively growing generative adversarial networks. So the ProGAN generator takes the inputs and keeps generating images along with a condition, whereas the discriminator keeps classifying the real and fake images; again estimates the average color in each generation process.

The study showed the logical process of the architecture along with mathematical expression used for consistency per these conditional inputs. For result comparison, additional two training experiments were performed on a style-based generative adversarial network and its conditional version with the same dataset training configurations. Improved results were obtained in all the three training results, including the proposed architecture as shown in the result and appendices section.

The experimental results in the proposed solution show that the fashion design can be done through a deep generative model algorithm by controlling important attributes during the generation. The model evaluation results in Frechet inception distance and Perceptual path

length metrics have gained significant values of FID50k 41 and PPL90k 1500, respectively. In both cases, the proposed architecture shows a better evaluation result. A human evaluation user study was performed to assess the capability of identifying the original and generated images with a confusion of the user 47% for paired and 46 % for unpaired evaluation setting. Therefore, this study shows that it is feasible to use such techniques to reduce human effort and manual works.

7.2. Future Work

The proposed generative adversarial network-based fashion design is limited to only color, shape, and texture. But the reality includes multiple and complex texture or patterns, size, and printing shape, and text. Fashion image generation with multiple conditional inputs can be further improved by adding more complex patterns, texts, and object shape that meets the reality design consideration. An architecture optimization for training stability is required and further, the model needs to generate a high-quality image that allows multiple conditional inputs. Creating a fusion of an algorithm and combine the potential benefits of the individual architecture. Moreover, as per the user study texture and shape, inconsistent issues were observed and needed to be addressed in future research.

SPECIAL ACKNOWLEDGMENTS

This research project is funded by Adama Science and Technology University under the grant number:

ASTU/SM-R/085/19

Adama, Ethiopia

REFERENCES

[1] J. U. Ahmed, M. H. K. Chowdhury, M. J. Uddin, and M. M. Ferdous, "Sadakalo: Marketing of traditional fashion in the modern fashion industry," *Vis. J. Bus. Perspect.*, vol. 18, no. 2, pp. 125–135, 2014.

[2] G. Yildirim, C. Seward, and U. Bergmann, "Disentangling Multiple Conditional Inputs in GANs," *arXiv [cs.CV]*, 2018.

[3] L. Liu, H. Zhang, Y. Ji, and Q. M. Jonathan Wu, "Toward AI fashion design: An Attribute-GAN model for clothing match," *Neurocomputing*, vol. 341, pp. 156–167, 2019.

[4] C. Yu, Y. Hu, Y. Chen, and B. Zeng, "Personalized Fashion Design," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[5] R. H. Banerjee, A. Rajagopal, N. Jha, A. Patro, and A. Rajan, "Let AI clothe you: Diversified fashion generation," in *Computer Vision – ACCV 2018 Workshops*, Cham: Springer International Publishing, 2019, pp. 75–87.

[6] www. fibre2fashion.com, "Fashion Designing History | Evolution of Fashion Designing | Growth of Fashion Design Industry | Fibre2fashion.Com," *Fibre2fashion.com*. [Online]. Available: https://www.fibre2fashion.com/industry-article/3730/fashion-designing-the-then-and. [Accessed: 23-Jul-2020] .

[7] "Definition of Fashion Designing," *Chron.com*. [Online]. Available: https://work.chron.com/definition-fashion-designing-25262.html. [Accessed: 23-Jul-2020].

[8] G. Yildirim, N. Jetchev, R. Vollgraf, and U. Bergmann, "Generating high-resolution fashion model images wearing custom outfits," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019.

[9] X. Han *et al.*, "Automatic spatially-aware fashion concept discovery," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[10] W.-C. Kang, C. Fang, Z. Wang, and J. McAuley, "Visually-aware fashion recommendation and design with generative image models," in *2017 IEEE International Conference on Data Mining (ICDM)*, 2017.

[11] O. Sbai, M. Elhoseiny, A. Bordes, Y. LeCun, and C. Couprie, "DesIGN: Design Inspiration from Generative Networks," in *Lecture Notes in Computer Science*, Cham: Springer International Publishing, 2019, pp. 37–44.

[12] W. Xian *et al.*, “TextureGAN: Controlling deep image synthesis with texture patches,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[13] S. Zhu, S. Fidler, R. Urtasun, D. Lin, and C. C. Loy, “Be your own Prada: Fashion Synthesis with structural coherence,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[14] W.-L. Hsiao, I. Katsman, C.-Y. Wu, D. Parikh, and K. Grauman, “Fashion++: Minimal edits for outfit improvement,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[15] M. Hargrave, “How deep learning can help prevent financial fraud,” *Investopedia.com*, 23-Jul-2020. [Online]. Available: https://www.investopedia.com/terms/d/deep-learning.asp. [Accessed: 23-Jul-2020] .

[16] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” *arXiv [stat.ML]*, 2014 .

[17] K. Vaccaro, T. Agarwalla, S. Shivakumar, and R. Kumar, “Designing the future of personal fashion,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI ’18*, 2018.

[18] Y. R. Cui, Q. Liu, C. Y. Gao, and Z. Su, “FashionGAN: Display your fashion design using Conditional Generative Adversarial Nets,” *Comput. Graph. Forum*, vol. 37, no. 7, pp. 109–119, 2018.

[19] M. Mirza and S. Osindero, “Conditional Generative Adversarial Nets,” *arXiv [cs.LG]*, 2014.

[20] Z. Al-Halah, R. Stiefelhagen, and K. Grauman, “Fashion forward: Forecasting visual style in fashion,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[21] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[22] C. Oeldorf and G. Spanakis, “LoGANv2: Conditional style-based logo generation with generative adversarial networks,” in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019.

[23] Z. Yu, J. Lian, A. Mahmoody, G. Liu, and X. Xie, “Adaptive user modeling with long and short-term preferences for personalized recommendation,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[24] A. Workie, R. Sharma, Y. K. Chung, and Adama Science and technology university, "Digital video summarization techniques: A survey," *Int. J. Eng. Res. Technol. (Ahmedabad)*, vol. V9, no. 01, 2020.

[25] K. E. Ak, A. A. Kassim, J. H. Lim, and J. Y. Tham, "Learning attribute representations with localization for flexible fashion search," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[26] J.-C. Wu, J. A. S. Rodríguez, and H. J. C. Pampín, "Session-based complementary fashion recommendations," *arXiv [cs.IR]*, 2019.

[27] Y. Ge, R. Zhang, X. Wang, X. Tang, and P. Luo, "DeepFashion2: A versatile benchmark for detection, pose estimation, segmentation, and re-identification of clothing images," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[28] C. Lassner, G. Pons-Moll, and P. V. Gehler, "A generative model of people in clothing," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[29] N. Kato, H. Osone, K. Oomori, C. W. Ooi, and Y. Ochiai, "GANs-based clothes design: Patternmaker is all you need to design clothing," in *Proceedings of the 10th Augmented Human International Conference 2019 on - AH2019*, 2019.

[30] J. Marcelino, J. Faria, L. Baía, and R. G. Sousa, "A hierarchical deep learning natural language parser for fashion," *arXiv [cs.IR]*, 2018.

[31] W.-L. Hsiao and K. Grauman, "Creating capsule wardrobes from fashion images," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[32] E. F. Johnson, "Defining fashion: Interpreting the scope of the design piracy prohibition act," *Brooklyn Law Rev.*, vol. 73, no. 2, p. 6, 2008.

[33] C. Lucey, "Ivanka Trump promotes women's empowerment in Ethiopia," *Associated Press*, 14-Apr-2019. [Online]. Available: https://apnews.com/9c9da630ddb7444c8db04fe27e6cce19.[Accessed:01-Oct-2020].

[34] A. Raj, P. Sangkloy, H. Chang, J. Hays, D. Ceylan, and J. Lu, "SwapNet: Image Based Garment Transfer," in *Computer Vision – ECCV 2018*, Cham: Springer International Publishing, 2018, pp. 679–695.

[35] W. Wang, W. Wang, Y. Xu, J. Shen, and S.-C. Zhu, "Attentive fashion grammar network for fashion landmark detection and clothing category classification," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[36] J. Zhu, Y. Yang, J. Cao, and E. C. F. Mei, “New product design with popular fashion style discovery using machine learning,” in *Artificial Intelligence on Fashion and Textiles*, Cham: Springer International Publishing, 2019, pp. 121–128.

[37] “Report for: A framework for generative product design powered by deep learning and artificial intelligence : Applied on everyday products,” *Altmetric.com*. [Online]. Available: https://www.altmetric.com/details/52204923. [Accessed: 23-Jul-2020].

[38] Z. Yang, Z. Su, Y. Yang, and G. Lin, “From recommendation to generation: A novel fashion clothing advising framework,” in *2018 7th International Conference on Digital Home (ICDH)*, 2018.

[39] C. Packer, J. McAuley, and A. Ramisa, “Visually-aware personalized recommendation using interpretable image representations,” *arXiv [cs.CV]*, 2018.

[40] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, “Recent progress on generative adversarial networks (GANs): A survey,” *IEEE Access*, vol. 7, pp. 36322–36333, 2019 .

[41] J. Feng *et al.*, “Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification,” *Remote Sens. (Basel)*, vol. 12, no. 7, p. 1149, 2020.

[42] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[43] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[44] M. A. Hobley and V. A. Prisacariu, “Say yes to the dress: Shape and style transfer using conditional GANs,” in *Computer Vision – ACCV 2018*, Cham: Springer International Publishing, 2019, pp. 135–149.

[45] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan, “Bidirectional conditional generative adversarial networks,” in *Computer Vision – ACCV 2018*, Cham: Springer International Publishing, 2019, pp. 216–232.

[46] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[47] W. Xian *et al.*, “TextureGAN: Controlling deep image synthesis with texture patches,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[48] Y. Nagano and Y. Kikuta, “SRGAN for super-resolving low-resolution food images,” in *Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management - CEA/MADiMa ’18*, 2018.

[49] C. Ledig *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” *arXiv [cs.CV]*, 2016.

[50] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” *arXiv [cs.NE]*, 2017.

[51] L. A. Gatys, A. S. Ecker, and M. Bethge, “A neural algorithm of artistic style,” *arXiv [cs.CV]*, 2015.

[52] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of StyleGAN,” *arXiv [cs.CV]*, 2019.

[53] N. Pandey and A. Savakis, “Poly-GAN: Multi-Conditioned GAN for Fashion Synthesis,” *arXiv [cs.CV]*, 2019.

[54] Tensorflow, “An open-source machine learning framework for everyone.” [Online]. Available: https://www.tensorflow.org/ .

[55] OpenCV.org, “Opencv.” [O nline]. Available: https://opencv.org/about.html. [Accessed: 27-Oct-2018].

[56] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs trained by a two time-scale update rule converge to a local Nash equilibrium,” *arXiv [cs.LG]*, 2017.

[57] K. Ak, A. Kassim, J.-H. Lim, and J. Y. Tham, “Attribute manipulation generative adversarial networks for fashion images,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[58] Y. Kwon, S. Kim, D. Yoo, and S.-E. Yoon, “Coarse-to-fine clothing image generation with progressively constructed conditional GAN,” in *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2019.

[59] S. Jiang and Y. Fu, “Fashion Style Generator,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017.

[60] N. Rostamzadeh *et al.*, “Fashion-gen: The generative fashion dataset and challenge,” *arXiv [stat.ML]*, 2018.

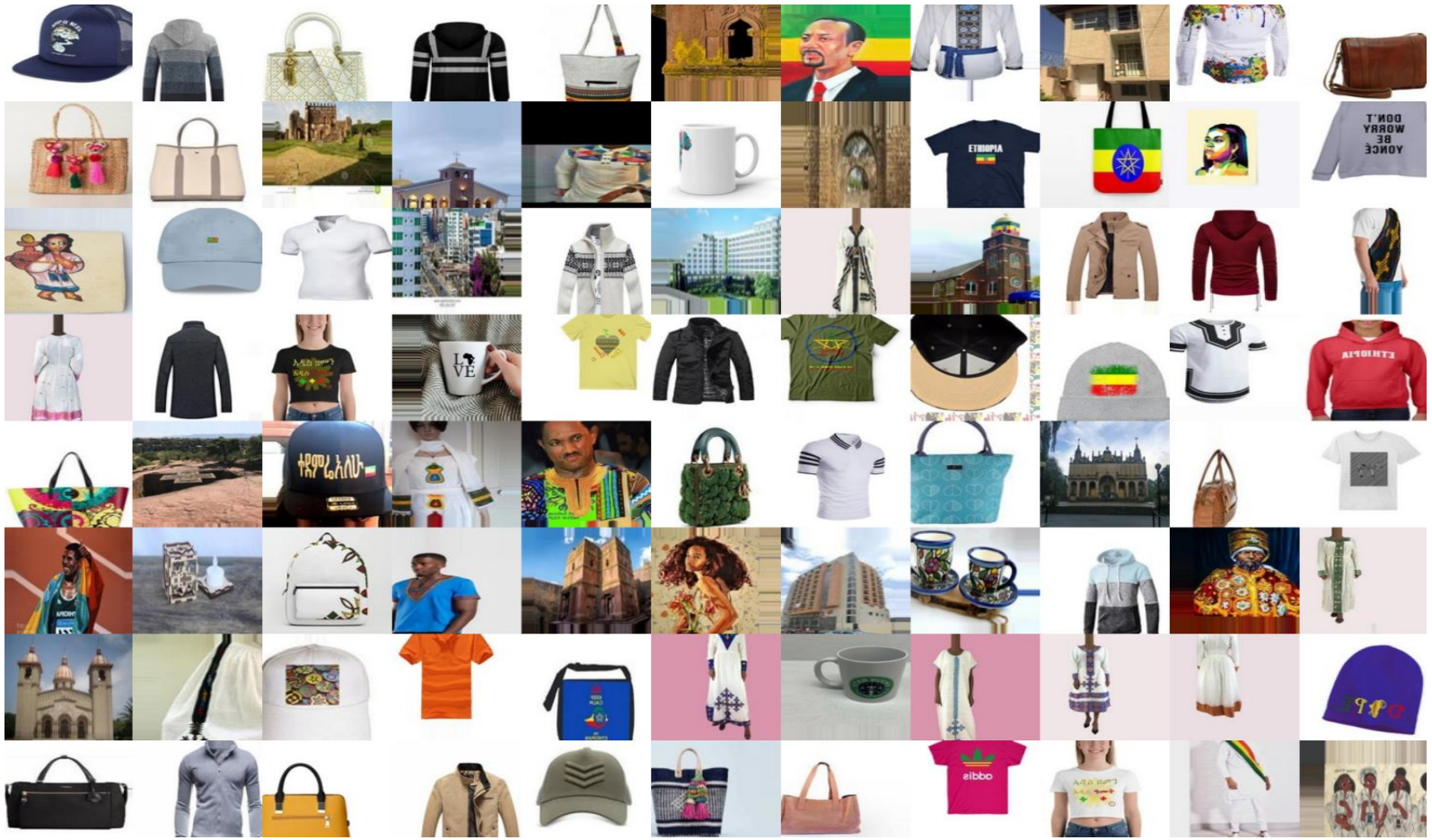
[61] X. Yang, S. Yuan, and Y. Tian, “Assistive clothing pattern recognition for visually impaired people,” *IEEE Trans. Hum. Mach. Syst.*, vol. 44, no. 2, pp. 234–243, 2014.

[62] Q. Ping, B. Wu, W. Ding, and J. Yuan, “Fashion-AttGAN: Attribute-aware fashion editing with multi-objective GAN,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[63] Akira, “From GAN basic to StyleGAN2 - Analytics Vidhya - Medium,” *Analytics Vidhya*, 22-Dec-2019. [Online]. Available: https://medium.com/analytics-vidhya/from-gan-basic-to-stylegan2-680add7abe82. [Accessed: 21-Oct-2020].

APPENDIXES

Appendix A: Orginal Sample Dataset



Appendix B: Sample Results from the Research

Appendix B.1 Training Network Summary

Initializing TensorFlow...

Running train.train_progressive_gan()s...

Streaming data using dataset.TFRecordDataset...

Dataset shape = [3, 128, 128]

Dynamic range = [0, 255]

Label size (conditions) = 3

Constructing networks...

Building TensorFlow graph...

This is a training summary for inputs, shape, and condition of the network.

Appendix B.2 Sample Code for Interactive Generation

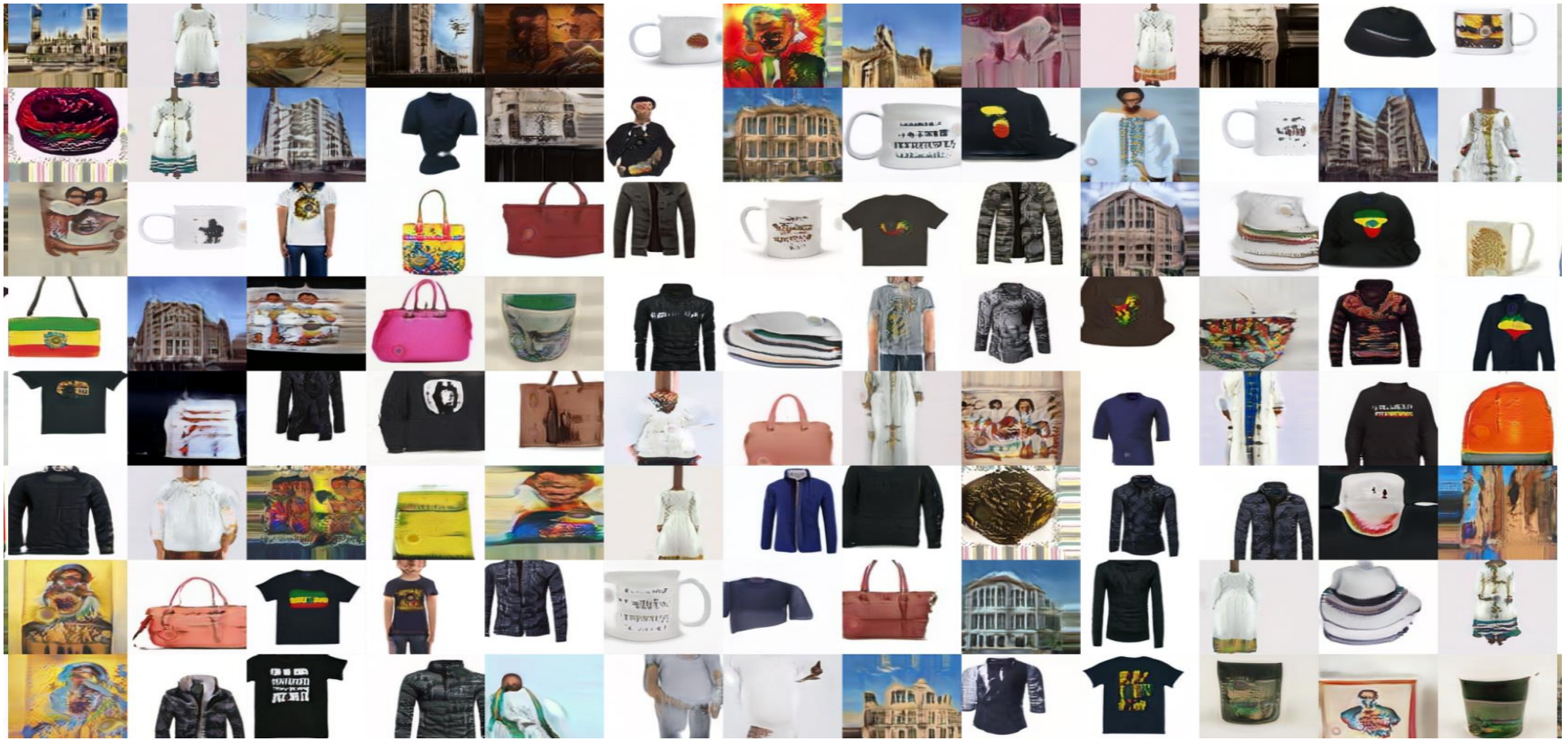
```
resume_network_pkl = "./results/model.pkl"
with tf.device('/gpu:0'):
    G, D, Gs = misc.load_pkl(resume_network_pkl)
imsize = Gs.output_shape[-1]
#------------------------
#Sample Random Shapes from mask
mask_list = glob.glob("./dataset/masks/*.jpg")#binary masks

def get_random_mask(batch_size):
    ix =np.random.randint(len(mask_list),size=(batch_size,))

    random_masks = []
    for i in ix:
        temp = Image.open(mask_list[i])
        temp = temp.resize((imsize, imsize))
        temp = (np.float32(temp) - 127.5)/127.5
        temp = temp.reshape((1, 1, imsize, imsize))
        random_masks.append(temp)
    random_masks = np.vstack(random_masks)
    return random_masks
def get_random_color(batch_size):
    return np.random.rand(batch_size, 3) * 2 - 1
def convert_to_image(x):
    return x.transpose((0,2,3,1)).clip(-1, 1) * 0.5 + 0.5
selected_shape = get_random_mask(1)
```

```
z1 = misc.random_latents(1, Gs)
z2 = misc.random_latents(1, Gs)
N = 2
fig = plt.figure(figsize=(N*6,6))
ax = []
art = []
for i in range(N):
    ax+=[fig.add_subplot(1,N,i+1)]
    ax[-1].axis('off')
    art += [ax[-1].imshow(np.zeros((1,1)))]

def f(r, z):

    r = r[1:]
    selected_color = np.array([[int(r[i:i+2], 16) for i in
[0, 2, 4]]])
    selected_color = (np.float32(selected_color) - 127.5) /
127.5
   selected_texture = z1 + z * (z2 - z1)
    st = time.time()
    GI = Gs.run(selected_texture, selected_color,
selected_shape)
    et = time.time()
    GI = convert_to_image(GI)
art[0].set_array(GI[0])
    ax[0].set_title(f"Inference Time: {(et-st)*1000:.0f}
ms")
    art[0].autoscale()

    art[1].set_array(selected_shape[0,0])
    art[1].autoscale()

interactive_plot = interactive(f,

r=widgets.ColorPicker(concise=False, description='Pick a
color', value='#aa00cc', disabled=False),

z=widgets.FloatSlider(min=0.0,max=1.0,step=0.1,value=0.0),
                                )

output = interactive_plot.children[-1]
output.layout.height = '500px'

interactive_plot

#color change
selected_textures = misc.random_latents(1, Gs).repeat(3, 0)

selected_shapes = get_random_mask(1).repeat(3, 0)

selected_colors = get_random_color(3)
```

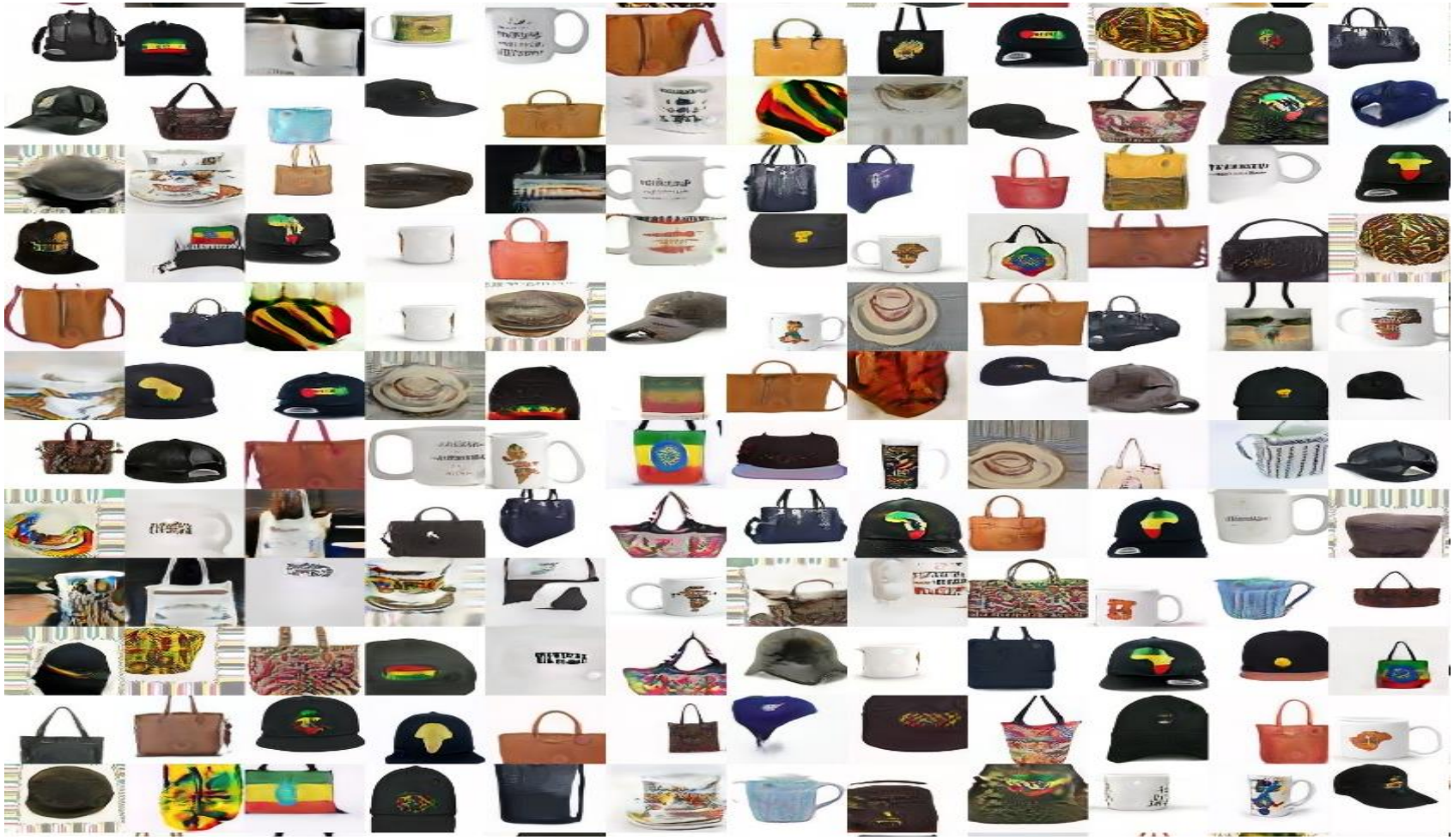
```
fake_images = Gs.run(selected_textures, selected_colors,
selected_shapes)

fake_images = convert_to_image(fake_images)

plt.figure(figsize=(12,4))
for i in range(3):
    plt.subplot(1,4,i+1)
    plt.imshow(fake_images[i])
    plt.axis('off')

plt.subplot(1,4,4)
plt.imshow(selected_shapes[0, 0], cmap='gray', vmin=0.0,
vmax=1.0)
#texture change
selected_textures = misc.random_latents(3, Gs)

selected_colors = get_random_color(1).repeat(3, 0)

selected_shapes = get_random_mask(1).repeat(3, 0)

fake_images = Gs.run(selected_textures, selected_colors,
selected_shapes)

fake_images = convert_to_image(fake_images)


selected_textures = misc.random_latents(1, Gs).repeat(3, 0)

selected_colors = get_random_color(1).repeat(3, 0)

selected_shapes = get_random_mask(3)

fake_images = Gs.run(selected_textures, selected_colors,
selected_shapes)

fake_images = convert_to_image(fake_images)
plt.figure(figsize=(12,4))
for i in range(3):
    plt.subplot(1,4,i+1)
    plt.imshow(fake_images[i])
    plt.axis('off')
plt.subplot(1,4,4)
plt.imshow(selected_shapes[i,0])
```

Appendix C: Sample Results from the Training

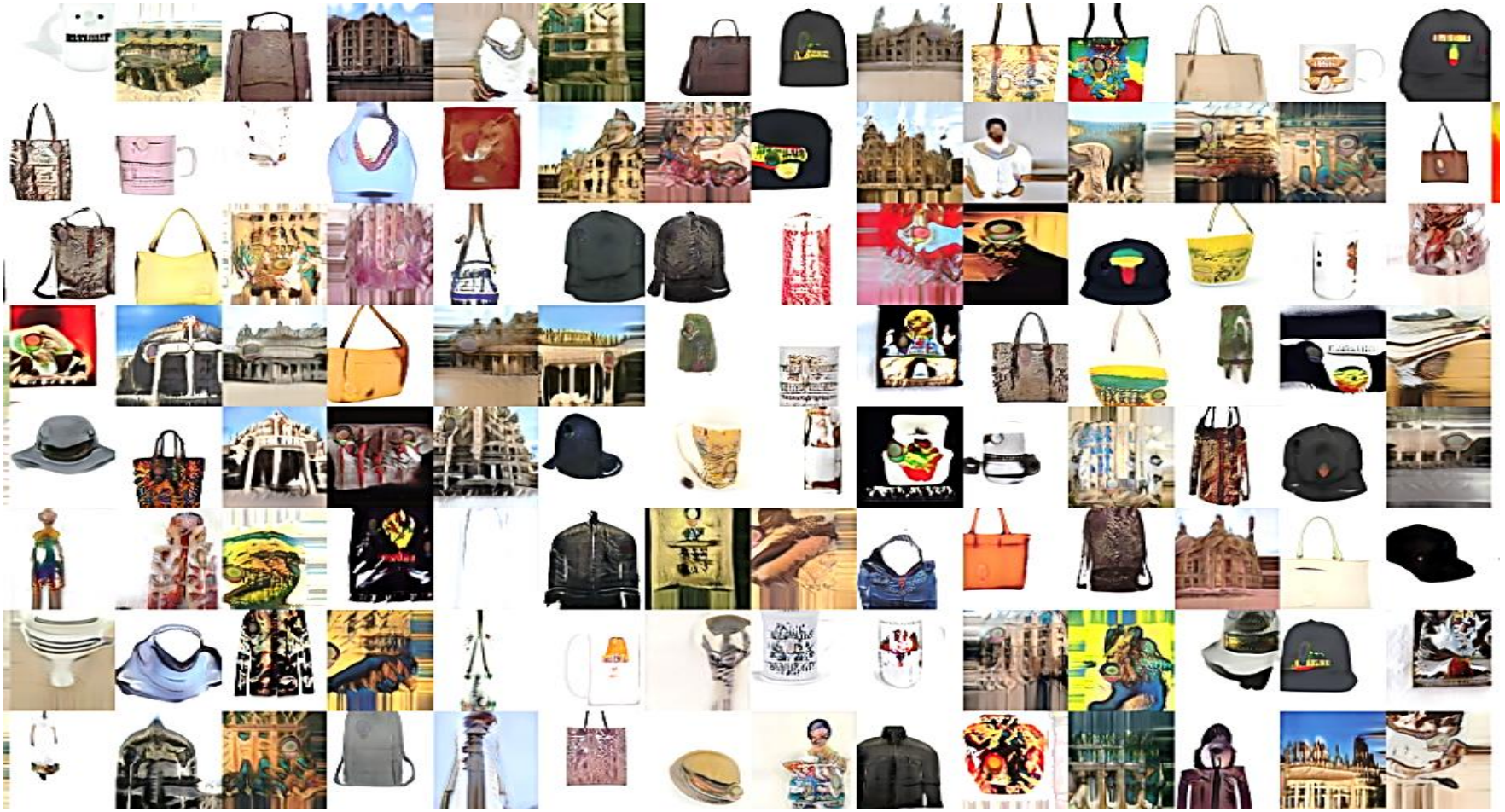
Appendix C.1 Result from Proposed Solution(Conditional ProGAN)



Appendix C.2 Result from StyleGAN



Appendix C.3 Result from Conditional Style GAN



Appendix D: Model Evaluation

Appendix D.1 Perceptual Path Length Model Evaluation

Condition Progressive GAN in different interpolation				
Experiments	***PPL_zfull***	***PPL_Wfull***	***PPL_zend***	***PPL_wend***
Perceptual path length	1500	640	1580	451
Time to complete evaluation	27 m 41sec	26 m 06 sec	26 m 29 sec	26 m 23 sec
StyleGAN				
Perceptual path length	1966.52	1011.30	1968.88	552.56
Time to complete evaluation	31m 21 sec	30 m 05 sec	30m 41 sec	30m 39sec
Condition StyleGAN				
Perceptual path length	2234.6	789.4	2240.2	421.2
Time to complete evaluation	24 m 55 sec	25 m 43 sec	25 m 27 sec	24 m 57 sec

Appendix D.2 Iterative Training FID Evaluation for Condition Progressive GAN

Checkpoint or model	***Time compute evaluation***	***Fid50K***
network-snapshot-000140	time 4m 40s	fid50k 358.6505
network-snapshot-001283	time 4m 42s	fid50k 269.1465
network-snapshot-002364	time 5m 36s	fid50k 241.1961
network-snapshot-003285	time 7m 11s	fid50k 210.0087
network-snapshot-004085	time 7m 13s	fid50k 143.7628
network-snapshot-004705	time 16m 00s	fid50k 54.5739
network-snapshot-005306	time 11m 47s	fid50k 41.5336
network-snapshot-005726	time 17m 02s	fid50k 42.0443
network-snapshot-006126	time 16m 14s	fid50k 46.6345

Appendix E: Human Evaluation User Study

Appendix E.1 Paired Human Evaluation User Study

Responses cannot be edited

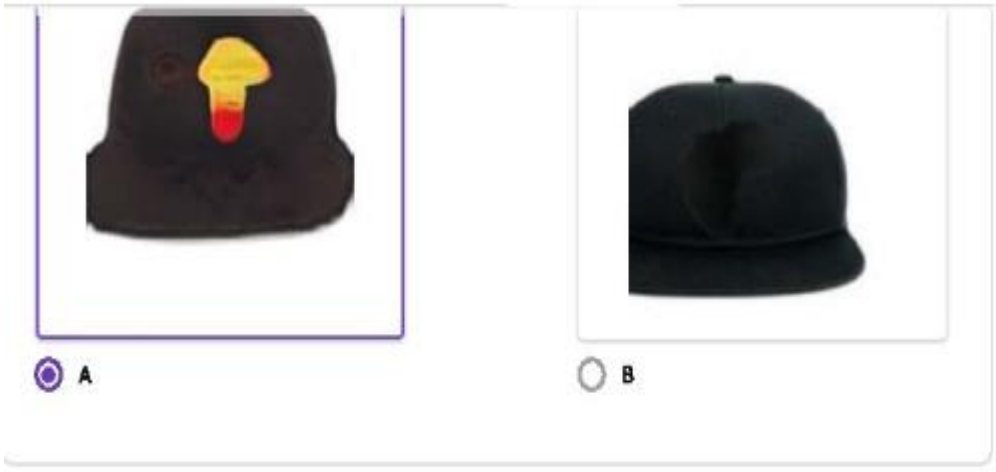
Human Evaluation User study

Thank You for participating in our questionnaires
paired user study

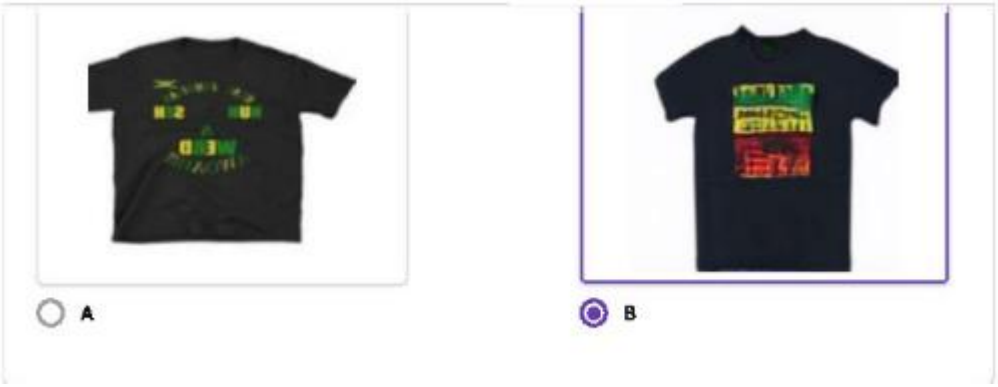
* Required

1) which of the following jacket is more realistic and more natural? *

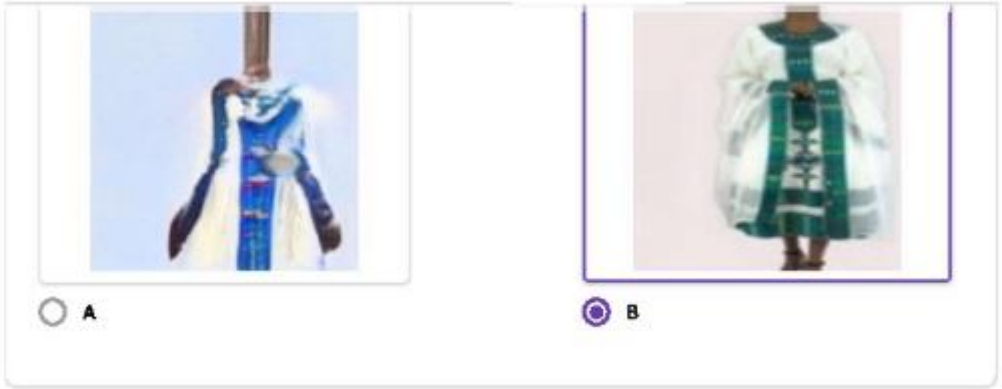
2) which of the following jacket is more realistic and more natural? *



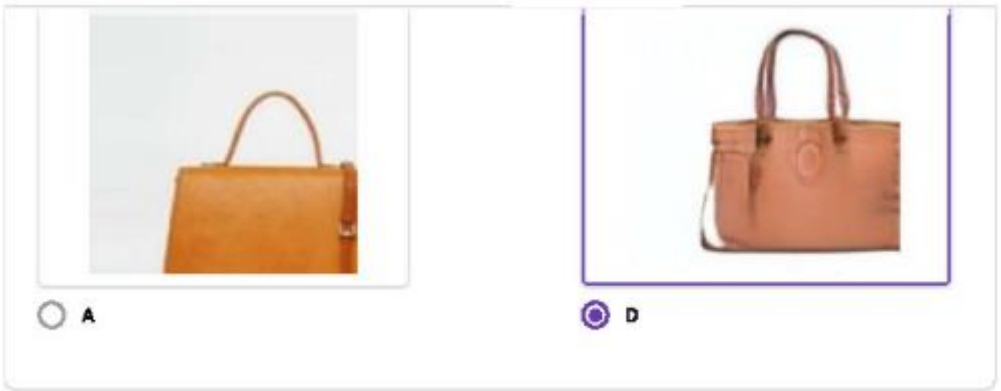
3) which of the following jacket is more realistic and more natural? *



4) which of the following jacket is more realistic and more natural? *



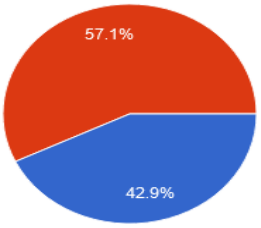
5) which of the following jacket is more realistic and more natural? *



5) Which following is more realistic and more natural? *

A

B



Answer for the above questions

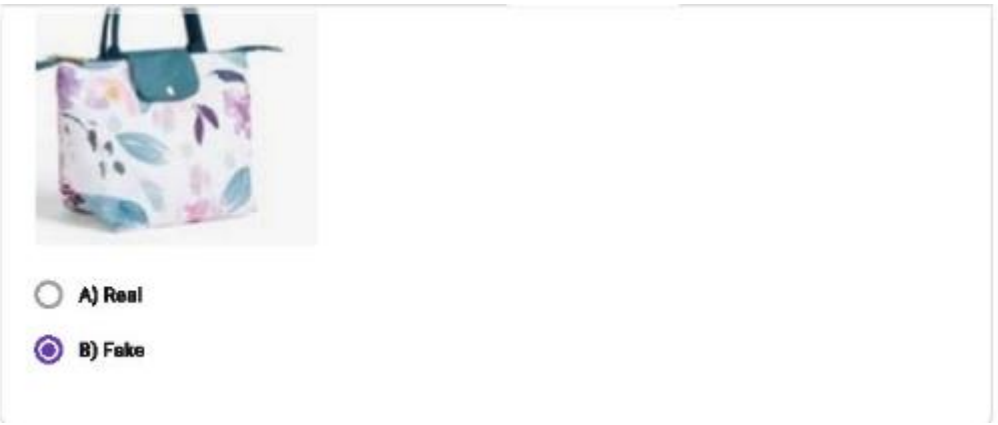
According to the response of 13 users in paired setting

		Paired human evaluation questions					
S.No	***Name user***	***Q#1***	***Q#2***	***Q#3***	***Q#4***	***Q#5***	***Sum /total***
1	Amare	B	A	B	B	B	
2	Anteneh	B	B	A	B	A	
3	Bereket	B	B	A	B	A	
4	Dagmawit	B	A	B	B	A	
5	Daniel	B	A	B	B	B	
6	Driba	B	B	A	B	A	
7	Habib	B	B	B	B	B	
8	Kirubel	B	A	B	A	B	
9	Minlik	B	B	B	B	B	
10	Natnael	B	B	B	B	A	
11	Gebre	A	A	B	A	B	
12	Tena	A	B	A	A	B	
13	Lidiya	A	A	B	A	B	
		Summary of the evaluation					
	Actual	B	B	A	B	A	
	Correctly	10	7	4	9	5	35/13
	Correctly/total	10/13	7/13	4/13	9/13	5/13	(2.69)/5=0.538
	Incorrectly	3	6	9	4	8	(30/13)
	Incorrect /Total	3/13	6/13	9/13	4/13	8/13	(2.307)/5 =0.46

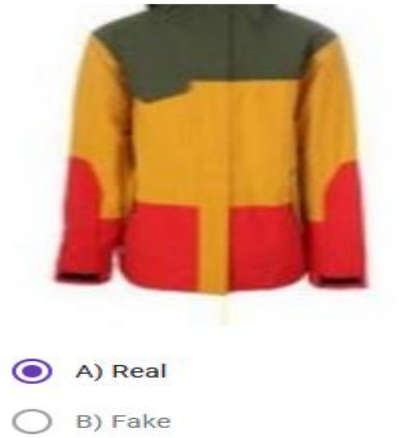
Key **A** Real B Fake

Appendix E.2 Unpaired Human Evaluation User Study

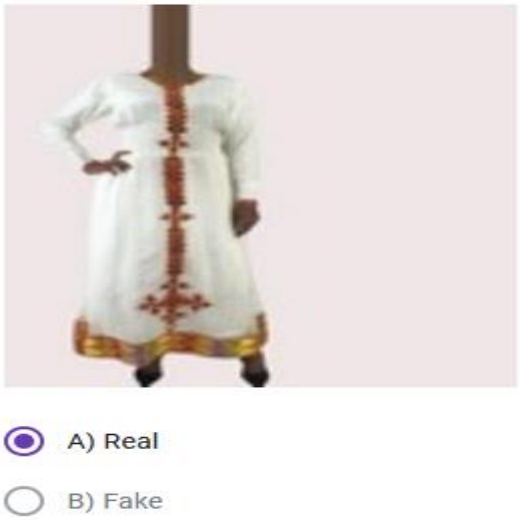
1) Look the following image and label it real if it is more natural and label fake if not *



2) Look the following image and label it real if it is more natural and label fake if not *



3) Look the following image and label it real if it is more natural and label fake if not *



4) Look the following image and label it real if it is more natural and label fake if not *

- (●) A) Real
- ( ) B) Fake

5) Look the following image and label it real if it is more natural and label fake if not *

- (●) A) Real
- ( ) B) Fake

Answer for the above questions

According to the response of 13 users in unpaired setting

		Question number					***Sum/total***
S.N	Evaluator Name	1	2	3	4	5	
1	Amare	B	A	A	A	B	
2	Anteneh	A	A	A	A	A	
3	Bereket	B	A	A	B	B	
4	Dagmawit	A	B	A	B	A	
5	Daniel	B	B	A	A	B	
6	Driba	B	A	B	B	B	
7	Habib	A	B	A	B	B	
8	Kirubel	A	A	A	B	B	
9	Minlik	A	B	A	B	B	
10	Natnael	A	A	B	A	A	
11	Gebre	B	A	B	A	A	
12	Tena	B	B	B	B	A	
13	Lidiya	B	B	B	A	A	
	Summary of the evaluation						
	Actual	A	A	A	A	**B**	
	correctly	6	7	8	6	7	34/13=2.61
	Correctly / Total user	6/13	7/13	8/13	6/13	7/13	2.61/5=0.52
	incorrectly	7	6	5	7	6	31/13=2.38
	Incorrectly /Total user	7/13	5/13	5/13	7/13	6/13	2.38/5=0.476

Key

A Real B Fake

Appendix E.3 Human Evaluation Quality Assessments

This is a detailed report for the user's response in quality assessments where images were given in random class to evaluate the texture, shape, color and beauties of images using scale range from 1-5 where values 1-2 for poor, 2-3 for medium poor , 3-4 for good,4-5 for very good and 5 for an excellent quality in every criteria mentioned as in the following table.

1) Conditional ProGAN results

1	2	3	4	5
○	○	○	○	○

2) StyleGAN results

1	2	3	4	5
○	○	○	○	○

3) Conditional StyleGAN results

1 2 3 4 5

Questions for the above question

1) Look at following image give a grade its color correctness
2) Look at following image give a grade its texture correctness
3) Look at following image give a grade its shape correctness
4) Look at following image give a grade its beauty (aesthetics)

1) The 9 person's responses for the human quality assessments for Conditional ProGAN

Users	***Color correctness***	***Texture correctness***	***Shape correctness***	***Beauties***
1	4	3	3	2
2	4	3	3	3
3	4	3	4	3
4	2	3	3	2
5	2	3	2	3
6	4	3	5	2
7	3	4	4	4
8	3	2	4	2
9	4	2	4	3
Averages	3.333333	2.888889	3.555556	2.666667

The overall quality assesments of proposed solution(conditional progan) in all above criteria is 3.1 shows that it is good.

2) The 9 person's responses for the human quality assessments for StyleGAN

Users	***Color correctness***	***Texture correctness***	***Shape correctness***	***Beauties***
1	3	4	3	4
2	3	3	3	4
3	3	2	3	3
4	2	3	2	1
5	2	3	3	3
6	3	1	3	3
7	3	4	4	3
8	3	2	2	3
9	3	2	4	2
Averages	2.777778	2.666667	3	2.888889

The overall quality assesments of StyleGAN in all above criteria is 2.82 shows that it is medium poor.

3) The 9 person's response for the quality assessments Conditional StyleGAN

Users	***Color correctness***	***Texture correctness***	***Shape correctness***	***Beauties***
1	4	4	3	2
2	4	3	2	4
3	4	3	4	4
4	2	2	3	3
5	2	2	3	3
6	2	1	2	4
7	3	4	4	4
8	2	3	2	4
9	3	3	4	3
Averages	2.888889	2.777778	3	3.444444

The overall quality assesments of conditional StyleGAN in all above criteria is 3.02 shows that it is good.