Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm

Efa Tilahun Bekabil

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

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

Office of Graduate Studies

Adama Science and Technology University

September 2022

Adama, Ethiopia

Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm

Efa Tilahun Bekabil

Advisor

Dr. Mesfin Abebe

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

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

Office of Graduate Studies

Adama Science and Technology University

September 2022

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled "**Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm** "is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Efa Tilahun Bekabil _____ _____

Name Signature Date

Recommendation

I, the advisor of this thesis, hereby certify that I have read the revised version of the thesis entitled "**Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm**" prepared under my guidance by **Efa Tilahun Bekabil** submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science and Engineering. Therefore, I recommend the submission of a revised version of the thesis to the department following the applicable procedures.

Dr. Mesfin Abebe _____ _____

Major Advisor Signature Date

_____ _____ _____

Co-Advisor Signature Date

Approval page

I, the advisors of the thesis entitled "**Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm**" and developed by Efa Tilahun Bekabil, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Dr. Mesfin Abebe	_____	_____
Major Supervisor	Signature	Date
_____	_____	_____
Co-Supervisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Efa Tilahun Bekabil have read and evaluated the thesis entitled "**Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm**" and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

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

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

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

ACKNOWLEDGEMENT

First of all, I would like to thank God for keeping me safe under conditions I and other have no power to control on them.

Second of all, I would like to thank my advisor Dr. Mesfin Abebe for accepting me as his advisee and showing me directions, correcting me and supporting me to the completion of this work.

Lastly I would like to say thank you all ASTU communities, Communities around ASTU, my teachers, my families and friends for direct and indirect support because I would not be here today without their contribution and support.

CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURE

FIGURE PAGE

ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
CNN	Convolution Neural Network
CPU	Central Processing Unit
CycleGAN	Cycle consistent Generative Adversarial Network
DCGAN Deep	Convolution Generative Adversarial Network
FID	Fréchet Inception Distance
GAN	Generative Adversarial Networks
GPU	Graphical Processing Unit
IS	Inception Score
RAM	Random Access Memory
ResNet	Residual Network
ReLU	Rectified Linear Unit
SGAN	Standard Generative Adversarial Network
SSIM	Structural Similarity Index Matrix
VAE	Variation Auto encoders

ABSTRACT

This thesis report presents deep learning model developed for designing culture reflecting fashion. Handle image generation with complex structure and multicolored was the problem of previous works and because of that it is difficult to implement existing deep learning framework for culture reflecting fashion design. To solve the failures of previous work we have developed Custom assisting algorithm for image feature extraction and transformation that enabled us to treat content and style information or feature of image independently. Culture dataset is developed. The way of representing culture on fashion design using sketch and cultural pattern is designed and implemented. Our model used sketch for fashion design like human and controlling sketch according is handled and presented. At the end results gained from our model is discussed and evaluation with baseline is presented. Generally, this is the first work by which culture reflecting on fashion is controlled and enabled. In our work sketch of an image is used to control representation of cultural heritages on fashion and to control local structure of an image and color patterns are used to control representation of other cultural elements like emotion and feelings. Using our custom assisting algorithm, a lot of problems could be solved since it is generic feature transformation and extraction algorithm.

Key Words: - Generative Adversarial Networks, Culture Reflecting, Deep Learning, Artistic Fashion Design, Framework, Architecture.

CHAPTER 1

1 INTRODUCTION

1.1 Background of the Study

This paper introduces the work we have done on "Culture Reflecting Artistic Fashion Design using Deep Learning and Assisting Custom Algorithm".

Before giving single definition of the title let us defines each terms and or phrases contained in the title. Culture is anything of a particular society used to describe, identify and represent that society at a particular period of time. Culture can be categorized as material an non material, real and ideal, and visual and non-visual or verbal culture. Fashion is social-expressing or advocacy in a particular period of time like clothing's, religion, literature, language, etc. also fashion can be defined as material and concepts used by a particular society in specific period. But the term fashion is used to represent tangible or material things in this paper. The work artistic is used to represent creativity with is already the concept included in the fashion definition. A fashion design is the process of specification and visualization to be produced before production of fashion. So then culture reflecting fashion design is the process of designing fashion in the way that it can reflect culture of particular society in artistic way (Plate, 2015).

This work focuses on designing culture reflecting fashion using deep learning and custom algorithm that assists the model deep learning model used. So the main objective of this paper is to develop deep learning frame work and custom algorithm for assisting that enable us design fashion through which culture can be expressed in artistic way.

Fashion design and production are the main processes in fashion industries. To design and produce fashion we should know the way to do that, which are step by step procedures. Both fashion design and production can be done at individual level, small enterprise level and industry level using, manually or machine guided. Fashion production can be categorized as traditional fashion production or modern fashion production. Shimena is an example of tradition fashion production system known is Ethiopia.

To produce or make fashion item some on should have in mind how to produce the fashion item, manually machines should be made or configured in the way that it is comfortable for the process and also it is the same for industries. Most of machine guided or automated fashion industries capabilities and configuration of components and materials are very important. Computer is among those machines. In all cases meaning individual, manual and automated industries the knowledge about fashion production is very important and someone should know or learn how to do that.

The step by step or process of making fashion is called Algorithm is computer science. Steps of producing fashion can be expressed using natural language and human can learn from it while for computers it can be expressed in the form of algorithm using computer languages. The process of having those steps in mind is called learning, which is solely dominated by human being for a long period of time. Now days, thanks to Artificial Intelligent, computers are learning.

Before AI, steps of processes of doing something are encoded as an algorithm and computer software are developed using those algorithms and helping human in many ways like fashion design and production. Now days instead of using static algorithm, which is not flexible and time consuming for coding for each and every tasks, dynamic algorithms are being developed and enabling computer or machines to learn like human being does.

AI deals with giving ability of human being to machines like computers and robots. AI is combination of decision making, problem solving and reasoning that is computer based and robots based skills. As the name implies, AI in the fashion business typically completes jobs that would otherwise be completed by humans. This technical innovation is being leveraged in this industry to improve the fashion movement and provide people more choices in clothing that is both sustainable and flattering to them.

Deep learning applications in fashion can be divided these applications into three categories: 1) Fashion recognition at the low level, 2) Fashion comprehension at the mid-level, and 3) Fashion applications at the high level. The classification presented below is based on the primary objective of each study. Therefore, keep in mind that certain categories do overlap. Higher-level applications may be composed of numerous lower-level or mid-level applications, for example,

try-on apps may also include parsing, labeling, categorization, and detection (Seyed Omid Mohammadi, March 202).

Traditional cloths are the main way in which reflecting culture and that differentiate one culture from the other. But the way they differ from culture to culture is by types of color they are made from and the way in which those colors are arranged. Not only traditional cloths, jewelries, bids and other things like ornaments used with traditional cloths are also differing in colors and their arrangement from culture to culture. Even flags (national, regional, organizational flags), differ in such away. Among culture elements color is the major and which can be expressed through fashion. Not only this, the fear of the impact of fashion on culture could be partially solved. Even many traditional cloth designers are using color of culture of society and designing very interesting fashion (Debi Robersona, 2005).

Fashion in which color of a many cultures are reflected are getting attention currently. Specially, on cultural events, weeding ceremonies, religion based events and on demonstrations or conferences wearing fashions design in such way is being very common. Our country is an example for such activities. For example, Irrechaa, Mesqel, Ashanda and others are participating peoples in millions of almost every participant with such like fashion.

However, designing such type of fashion, which cultural elements reflected through, are very challenging and time consuming. Even, the need of culture reflecting fashions is not only on culture related events or ceremonies but also we are seeing many fashion with colors related with culture or flags. Fashion industries are turning to producing such like fashions. But the very challenging thing is that it is very difficult or even impossible to do it manually.

Generative Adversarial Networks are types of deep learning model currently doing incredible change in the fashion related tasks. But, the problem of handling multi-colored and structurally complex image generation still existing. By having this problem was very hard to generate fashion images in the way they can reflect culture which is basically result of art works passed from generation to generation. So we proposed and implemented culture reflecting fashion design by developing algorithms in order to handle existing problems of previous works. Generally, this thesis report presents model that have capability of designing fashions in the way

that they can reflect culture of a given society without losing their original structure and color composition including messages.

Our main contributions:

- ✓ New color feature extraction Technique.
- ✓ White Area pixel migration algorithm for shape compatibility of pattern image and objects in images
- ✓ New Technique for gray image intensity value transformation based on color feature
- ✓ Cultural Pattern dataset is built
- ✓ Deep learning framework capable of handling culture reflecting fashion image generation tasks with multiple color and complex texture representing content or local structure and color arrangement

1.2 Motivation of the Study

GAN models are very interesting area in computer vision and robotics and many of fashion design problems are being solved by GAN models. Fashions with colors those are related to a given culture are being used for culture related events and religion based events or ceremonies. For countries like Ethiopia, our country, of multi culture and in which events or ceremonies based on culture and or religion including such things in fashion design is very important for fashion industries and small group or individual entrepreneurs. We were motivated to implement GAN models to solve problems of fashion design by which culture can be reflected.

Another motivation is that, fashions with colors those are related to a given culture are being used for culture related events and religion based events or ceremonies. For countries like Ethiopia, our country, of multi culture and in which events or ceremonies based on culture and or religion including such things in fashion design is very important for fashion industries and small group or individual entrepreneurs.

1.3 Statement of Problem

Traditional clothes have multiple color with complex structures and patterns. These complex and mixture of colors are used to represent culture and reflects and identify a given society. Designing fashions with such complex structure and mixture of multiple colors is the very challenging task for human. Generative Adversarial network is being used to design fashions including abstract representation of features on the fashion. But the existing GAN algorithms

could not able to design fashions with complex structure and multiple color mixtures like traditional clothes. Even though deep learning algorithms are generating diverse images for fashion design the style of generated image is constrained by single or uniform color or texture. In case of fashion image with complex structure and multiple color arrangements and mixture the output is very far from ground truth. Also feature transfer from one fashion to another fashion is remaining static and flexibility of these features is very challenging. Edges distortion, monotonicity of color or texture nature and gap between expected result and generated images are problems of previous works. By this paper I am presenting culture reflecting artistic fashion design that extract colors extract color information from traditional cloths, prepare complex arrangement and mixture of these colors and design fashions reflecting culture by using GAN deep learning algorithm and assisting custom algorithm

1.4 Research Questions

- How to develop deep learning model that design fashion that reflects culture in artistic way and that is modern using GAN deep learning algorithm and assisting custom algorithm.
- How to design and develop framework of Conditional Generative Adversarial network supported with custom algorithm that able to design fashion with complex patterns and multi colors.
- How to prepare datasets and preprocess for the model towards the goals of the study

1.5 Objectives of the Study

This topic explores the objectives of the study. The objectives of the study, general objective and specific objectives are presented in the following subtopics respectively.

1.5.1 General Objective

The main objective of the study was to develop a model for culture reflecting artistic fashion design with complex arrangement, structure and mixture of colors which could not be solved with existing deep learning frame works sing GAN deep learning algorithm and custom algorithm that supports GAN framework being proposed

1.5.2 Specific Objectives

- How to develop deep learning model that design fashion that reflects culture in artistic way and that is modern using GAN deep learning algorithm and assisting custom algorithm.
- How to design and develop framework of Conditional Generative Adversarial network supported with custom algorithm that able to design fashion with complex patterns and multi colors.
- How to prepare datasets and preprocess for the model towards the goals of the study

1.6 Scope and Limitation

1.6.1 Scope of the Study

The proposed model uses real image sketch, texture and traditional cloth color features as input. The color feature of the traditional cloth is mixed with the texture and the model learns from the reproduced texture and sketch of the real image to design the fashion.

1.6.2 Limitation of the Study

The followings are the limitation of the proposing model.

- ✓ The work does not include shape change during the design
- ✓ From the cultural or traditional cloth only features used are color features.
- ✓ White, black and or grayscale color features are not included as a color feature.

1.7 Applications of the Study

The work could be applicable in the following areas.

- ✓ Garment industries: garments design and produce different clothes, shoes, bags and other things. Especially garment industries those are interested in designing and producing culture based fashions will be benefited from the research.
- ✓ Individuals and small entrepreneurs those design and produce clothes, shoes and bags for culture based events could also be advantageous from the work
- ✓ Generally, countries those have multiple culture and multiple cultural ceremony or events, like our country could get many advantage from this work.

1.8 Organization of this Report

Chapter one of this thesis reports is to introduce the report and to remind ideas presented back in proposal. In chapter 2 is literature review and related works are presented. Methodologies used are presented in chapter 3. In chapter 4 data, models and algorithms including training and evaluations steps are designed and presented. Chapter 5 explores about implementation part of the work. Chapter 6 discusses and compares results gained. Conclusion and future directions are discussed in chapter 7 of this thesis report. References and appendices are putted at the end of this report.

CHAPTER 2

2 LITERATURE REVIEW AND RELATED WORKS

2.1 Overview Literature Review and Related Works

Under this chapter literatures reviewed and works those have direct relationship with our work are discussed. First general overview about AI, Machine learning, Deep learning and Generative Adversarial Networks are reviewed and their applications in fashion industries and fashion related tasks are reviewed. As related works we have studied related works by categorizing under three groups namely sketch based related works which means those focused on generating fashion image from sketch and other additional information, multiple input conditioned fashion generation and culture related fashion tasks. At the end of this chapter they way feature of fashion images are treated and measured by related works are discussed and summarized.

2.2 Literature review

2.2.1 Relationship between Culture, Fashion, Products and Industries

Culture is anthropological term that refers custom, beliefs, norms, values, languages, religions and other things used to identify one group of people from another. Fashion is dynamic and element of culture used to express a particular society in a particular time such as clothes. Linking culture with products has many advantages for fashion industries and consumers of their products. Clothes and fashions used as communications tools: in different parts of the world different society prefers fashion products their cultures are reflected on. Fashion industries are using culture as tool to sell products and in different society reflecting their culture using fashion is very common(Pozzo, 2020).

2.2.2 Ways of visualizing Culture through Fashions

When visualization or reflection of culture on fashion is concerned, knowing elements of culture is very important. (Introduction To Sociology: Understanding And Changing The Social World) categorized culture into two groups namely Material Cultures and Nonmaterial Cultures. Physical objects like clothes, technologies, tools and other tangible things in culture are

classified as material culture while nonphysical things like languages, belief, norms, symbols used for nonverbal communications, etc. are classified as nonmaterial culture.

Some of elements of culture can be represented or visualized in images. Material or tangible cultures can be represented using images like fashion items or clothes which are also elements of culture. Also there are some elements of nonmaterial culture which can be represented in image for example symbols used for nonverbal communication like body gestures and feelings or emotions (Plate, 2015).

Generally, representation or visualization of cultural elements on image can be possible using:

- ✓ Geometric shapes: cultural heritages like coins and other tangible elements of culture can be visualized or represented in images using geometric features or shapes(Ruggero Pintus, 2016)and (Riello, 2008).
- ✓ Color emotion or feeling which one of nonmaterial cultural elements is is can be represented using color. Color is used to express feelings or emotion and different colors have different meaning in different cultures in the world (Delwin T. Lindseya, 2008) and (Domicele Jonauskaite, 2020).
- ✓ Combination of geometric shapes and colors: combination of different geometric shapes, combination of different colors and or combination of geometric shapes and colors used to make cultural patterns that is among cultural elements. This type of culture representation is very common in African, Asian and Indian Cultures (Domicele Jonauskaite, 2020), (Riello, 2008) and (Ramirez, 2019).

2.2.3 Applications of AI in fashion

Applying AI, Machine learning, Deep learning in fashion is opening new opportunities for industries of fashions. Currently many of fashion related tasks are being solved using by AI. In Smart Fashion(Seyed Omid Mohammadi, March 202) categorized applications of AI, Machine learning and Deep learning in fashion industries into 9 namely classification, detection, virtual-try on, Fashion Retrieval, Fashion Syntheses, Recommendation, Analysis, Production, Feature extraction, Datasets and others. Even though these are the main application areas of AI in fashion they can be subcategorized into many groups and studied. For simplicity we focused on

application of Generative Adversarial Networks which are relatively close to our works and also sub category of Deep learning.

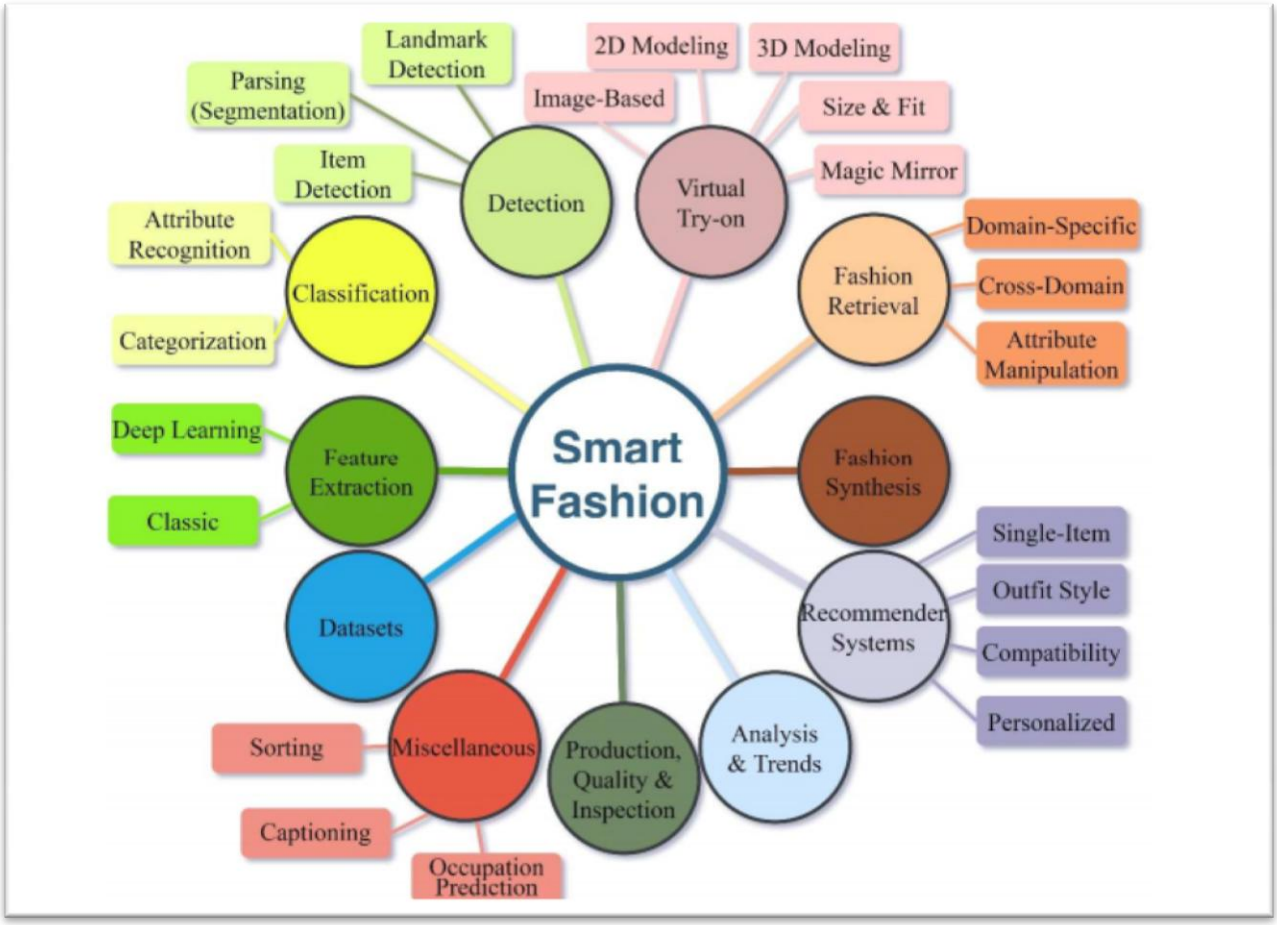


FIGURE 1) DIAGRAM OF SMART FASHION CATEGORIES (SEYED OMID MOHAMMADI, MARCH 2021)

Also many works are implemented in area of fashion using GANs Models. Some examples of GANs models developed and implemented around fashion problems are reviewed and Table 1) Applications Of Ai, Machine Learning, Deep Learning And Gans In Fashion Industries shows summary of them.

TABLE 1) APPLICATIONS OF AI, MACHINE LEARNING, DEEP LEARNING AND GANS IN FASHION INDUSTRIES

No	Tasks	Description	Examples
1	Fashion Retrieval	This task is about searching fashion items in database of image and retrieving it based on different features like visual features, content features and others. Domain-specific (when input and output are in the same domain), Cross-Domain (when input and output images are in different domain) and Attribute Manipulation (when someone needs the same fashion with different feature or with some change on it.) tasks are subtasks of fashion retrieval	Cross-domain image retrieval (J. Huang, 2015) Deepfashion (Z. Liu, 2016) ClothOut (Haijun Zhang, 2018)
2	Fashion Recommendation	Suggesting fashion items based on their similarity, style, color, and others. Single-Item recommendation, style or outfit recommendation, personalized recommendation and fashion compatibility recommendations are subtasks.	POG (Jiaming Xu, 2019) Visual Aware Recommendation (Wang-Cheng Kang, 2017)
3	Landmark detection	Is the task of finding fashion key points?	Landmark detection and classification (Thomas Ziegler, 2020)
4	Parsing Task	Is task of segmenting fashion items based on their semantic? Differ from fashion detection by presence of bounding box around the object in fashion detection task	Fashion Style (Kristen Vaccaro, 2016) Attr-GAN (Linlin Liu a, 12 March 2019)
5	Virtual Try-On	This task assists human to see how the fashion item looks like when they put it on using their image instead of try it physically Image based virtual try on task is 2-dimensional.	Dress Inorder (Aiyu Cui, 2021) Clothflow (Xintong Han, 2019)

		Modeling tasks: This task can be classified into 2D and 3D modeling. Fashion modeling is a task of synthesizing fashion image that enables view of that image from different angles. **Size and fit task:** This is task of predicting size of fashion item based on user's body shape and size.	
6	Fashion Classification	Classification is the task of determining class in which items are belonged. Classifying fashion items and or fashion items attributes or attribute recognition are fashion related tasks	Landmark detection and classification (Thomas Ziegler, 2020) Fashion Landmark Detection (Liu, 2016)
7	Fashion Feature Extraction	Fashion feature extraction is task of generating information about the fashion image which can be attributes of the fashion image in the way it can be described.	DeepFashion (Y. Ge, 2019) Toward AI fashion design: An Attribute-GAN model for clothing match (Linlin Liu a, 12 March 2019)
8	Fashion synthesis	Deals with synthesizing instance of fashion item image and designing from zero. Virtual try on and other tasks can be seen as image synthesis instead of differences of their application	ClothGAN (Qiang Wu, 2021) Structural based Systhesis (Shizhan Zhu)

2.3 Related works

2.3.1 Sketch based related works

2.3.1.1 Pix2Pix

FIGURE 2) EXAMPLES OF GENERATED IMAGES BY PIX2PIX

Pix2Pix (Efros, 2017) is general-purpose Image-to-Image Translation with Conditional Adversarial Networks model. In addition to sketch to image translation task pix2pix (Efros, 2017) generates photos from labels and image colorization. The goal of the pix2pix (Efros, 2017) was to develop a common framework for all these problems that predicts pixel from pixel. Using sketch image and random latent vector pix2pix (Efros, 2017) generates image with its structure is translated from sketch image and color and texture is form the random latent vector inputted. To control the structural similarity of generated image and reference image L1 or Mean Absolute Error loss is used in addition with adversarial loss. Pix2pix (Efros, 2017) uses U-Net(Olaf Ronneberger, 2015) architecture for generator and PatchGAN architecture for discriminator model.

Although it is general purpose model that solves some of problems from previous works, still there are things not been solved by Pix2pix (Efros, 2017). For example, latent vector inputted is intended to control the diversity of image generated in the form of its color and texture but it is not controlled by loss function and the direction of generated image diversity is not defined clearly.

2.3.1.2 TextureGAN

FIGURE 3) EXAMPLES OF IMAGES GENERATED BY TEXTUREGAN

Starting from problems of previous works, difficulties of controlling texture through style transfer methods and limitation of color controlling in previous works, RGB color space used is against semantic understanding of model that causes black and white color becomes seldom green proposed new GAN framework that generated realistic fashion image from sketch and texture patch arbitrarily attached on the sketch by controlling the propagation of that texture. TextureGAN takes sketch and patch of texture image attached to sketch at random location as input and generates fashion image of its structure translated from the sketch and its color and texture is to be translated from the texture patch. Its user level objective is to enable user to get realistic looking fashion image from sketch patch of texture is attached on it. At model level the objective is to control texture expansion or propagation through the part of sketch it is attached in. unlike Pix2Pix GAN in which color and texture of the generated image is determined by random latent space, in TextureGAN color and texture of generated image is determined by the texture patch attached on the sketch.

To implement their ideas authors used sketches of 256 X 256 and texture patches of 64X64 are extracted from handbags, shoes and clothes. Sketch images are paired with images it stands for but patches are stored separately without their origin. Their main focuses are feature extraction and improvements on loss functions.

Generally, to the followings are points are core or main ideas and contribution of TextureGAN.

- ✓ Lab color space is used instead of RGB, which is believed to go against semantic interpretation of the model.
- ✓ To test reality or fakeness prediction by discriminator model only L channel of images are used, this is because of additional characteristics to generated images resulted from texture attached on sketch input.
- ✓ Structure or content of generated images are to be controlled by the sketch input, so that to enforce the content of the generated image to be translated from sketch or to look like that of reference image two objective functions namely adversarial loss and feature loss are used. Adversarial loss is calculated from L channels of generated image and reference content image. Feature loss is to control high level semantic information and to enforce the generated image to have object structure specified by the sketch image. Feature loss is calculated from extracted features by VGG-19 from L channels of generated image and reference content image at relu_4_2 layer.
- ✓ To control the propagation of texture through the object three additional loss functions namely style loss, pixel loss and color loss are used. L channels are used for both style and pixel loss and *ab* channels are used to calculated color loss. Here by repeating L channels, Gramm matrices from pre-trained VGG-19 are extracted as texture features and used to calculate style loss. For color loss L2 loss calculated between generated and ground-truth.

2.3.1.3 FashionGAN

FIGURE 4) EXAMPLES OF IMAGES GENERATED BY FASHIONGAN

FashionGAN is Virtual Garment Display frame work using BicycleGAN architecture. Generates garment images from sketch or contour image and fabric pattern image. Users can get garment image having similar shape as sketch shape and texture similar to texture of fabric patterns they want.

Authors of FashionGAN analyzed solution proposed by privies works for the existing problem of CNN pre-defined loss function which is not suitable for generation of some realistic images and identified their failures and proposed new framework they believed to solve those failures. The followings are summary of those works.

- ✓ Pix2Pix: - problem of un deterministic control of texture and color
- ✓ BicycleGAN: - Problem of undetermined and inconsistency of visual information contained in latent vector.
- ✓ TextureGAN: - dependency of generated image texture on size and location of attached texture patch
- ✓ CycleGAN and MUNIT: - difficulties of generated images with detailed texture. Which means difficult to measure or control texture features of generated image caused by Unsupervised nature of the model.

They generalized as they cannot do one-to-one mapping between generated images texture with reference texture. As a result, they proposed new framework to handle the problem.

To handle the problem identified they focused on architecture and objective function or loss functions or the model. They have added Encoder model that takes fabric patterns as input and outputs latent vector containing only material and visual information of the fabric pattern. The output of the encoder model is fed to generator model with sketch image to generated last image with desired properties. The other contribution is that their first network takes sketch and fabric pattern extracted from reference image while the second network takes sketch and fabric pattern extracted from generated image instead of from original or reference image. the objective is to reach a state of model where fabric patterns extracted from original image and from generated image become similar according to their similarity measurement.

What makes their model architecturally and procedurally different from BicycleGAN is that, in BicycleGAN the first and second networks contains only one generator and discriminator which Encoder model is added in FashionGAN. Also in BicycleGAN the cycle is between domain A and domain B and vice versa, which means sketch to garment and garment to sketch and the like while in FashionGAN is between taking texture from original image or from generated image.

2.3.1.4 Summary of Sketch Based Related Works

When we analyze sketch based fashion generation tasks we have reviewed and studies their solutions and failures, there are common problems they share together or fail to succeed and problems belonged to each.

- ✓ Common problems/failures of TextureGAN and FashionGAN:
    - Color homogeneity: - their color contained in the texture patch or fabric pattern should be mono color or single color. Their model cannot learn multicolored pattern conditioned image generation or mapping.

- Problem of regular patterns: - when irregular patterns are inputted with sketch image generated images is colorless (black or white), missing parts, with no texture properties like the intended one.
- Effect of input features on generated images is not measured, because of this, properties of like texture features or patterns are combination of texture inputted and original image. if color or texture feature of reference image is appearing in generated image means their model is misinterpreting sketch as texture feature condition or translating color or texture of image from sketch, which is intended to be controlled by texture or pattern feature inputted.
- The same model is trained with different classes at different time. For example, to generated shoes from its sketch, handbags from its sketch and others differently which means single model cannot be used for handbag and shoes generation from their perspective sketch image.

✓ Pix2Pix: is not clear that random latent vector is only to control color and texture, and not controlled which leads to color similarity between generated image and reference image.

✓ Generally, because of these failures it is hard to generated images having cultural patterns and multiple colors.

2.3.2 Multiple Inputs Conditioned Related Works

2.3.2.1 DisentanglingGAN

FIGURE 5) EXAMPLE OF DISENTANGLING GAN GENERATED IMAGES (GÖKHAN YILDIRIM C. S., 2018)

Disentangling Multiple Conditional Inputs GAN (Gökhan Yildirim C. S., 2018) is Conditional Generative Adversarial Network proposed to control the effects of inputs on generated image. As a result, loss functions are defined in order to control the effects of different inputs by keeping some of inputs constants while the feature to be controlled is different. To demonstrate their work, they proposed conditional GAN conditioned with shape feature, 3D matrix RGB color space and texture feature. Shape feature and texture features are extracted from the same reference image. The RGB color generated randomly for controlling the color of fashion or article image intended to be generated. Color consistency, shape consistency, texture consistency loss functions are proposed as loss function to control the consistency of features of generated image at different time.

Even though some problems still remain unsolved. Generation of image when texture input extracted from complex texture and multicolored, article could not be handled. In some cases, color of generated images is different from their average color and from the inputted color feature. Drawing conclusion about color of the whole pixels in the image based on their average color leads the model to not to interpret the color feature inputted as color feature determinant or to determined color feature of generated image from other feature inputted with the color feature inputted. Also texture extraction mechanism they used Laplacian Matting Matrix included color information of the reference image as they form pattern which is in turn could be texture feature.

2.3.3 Culture Based Related Works

2.3.3.1 ClothGAN

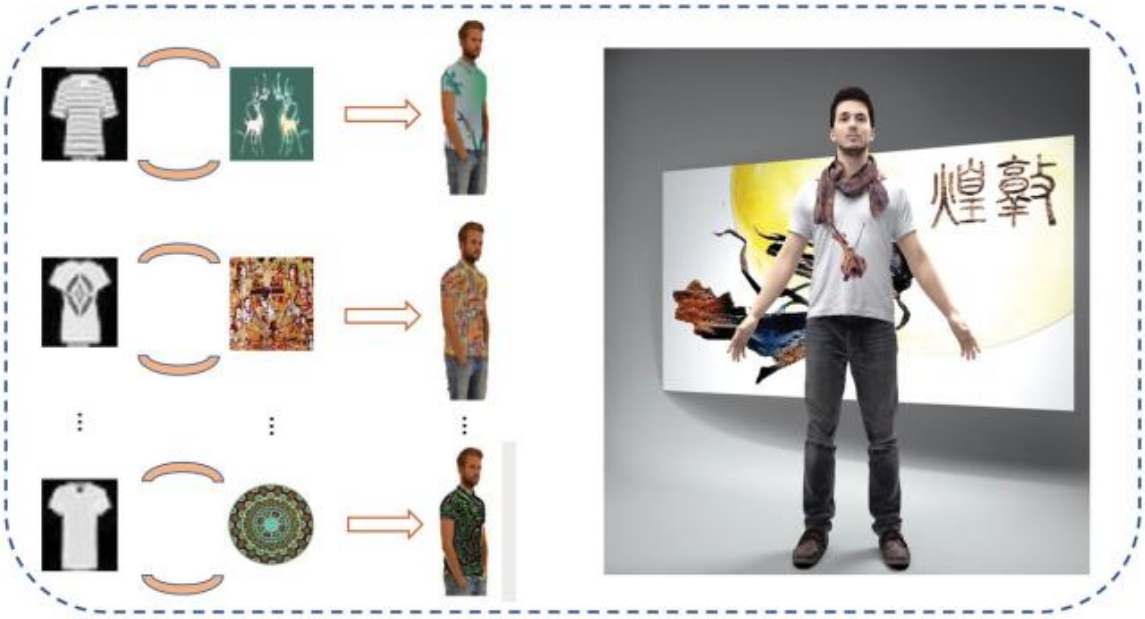


FIGURE 6) EXAMPLE FROM CLOTHGAN (QIANG WU, 2021)

By believing that clothing reflects pursuit of beauty using patterns and styles new conditional GAN framework called ClothGAN (Qiang Wu, 2021) is developed to generate cloth images reflecting cultural patterns exists in Mogao Caves at Dunhuang. The objective is to develop GAN model for Dunhuang clothing culture to design clothing for protecting and inheriting Dunhuang clothing culture, which is purpose specific or oriented model.

Fashion Mnist dataset and Dunhuang datasets containing cultural patterns images are used. For generation process generator module and style transfer module are developed. The generator takes images from Mnist dataset with random noise in order to generate multiple images of the same article. The output image from generator with cultural pattern image from dunguang dataset inputted to the style module.

Generators are conditioned with random latent and fashion image with additional constraints that represents class or type of image intended in word like jacket, pants or shirts. The generator models have their own discriminator for classifying generated images as real or fake. The one thing should be clear here is that the style transfer module is independent and no discriminator is provided

2.3.4 Image Feature Treatment and Related Works

Let say we have imaged $\boldsymbol{A}$ and image$\boldsymbol{B}$. Let $\boldsymbol{m}$ be mapping function, $\boldsymbol{f}$ be feature(s) extraction function, $\boldsymbol{\theta_m}$ be model $\boldsymbol{m}$ parameter (s), $\boldsymbol{z}$ be random latent information and $\widetilde{\boldsymbol{B}}$ image to be generated by model$\boldsymbol{m}$**.** Using this information let us formulate or represent some related works or reviewed works in the form of mathematical equation to study about how they treated features of images.

- ✓ Image generation from latent information

$$\tilde{B} = m(z,\theta_m) \quad \text{……………………………………………………}(2.1)$$

- ✓ Image generation from image A and latent information

$$\tilde{B} = m(A, z,\theta_m) \quad \text{……………………………………………………}(2.2)$$

- ✓ Image generation from features of image A and latent information

$$\tilde{B} = m(f(A), z,\theta_m) \quad \text{………………………………………………}(2.3)$$

- ✓ Image generation from image A

$$\tilde{B} = m(A,\theta_m) \quad \text{……………………………………………………}(2.4)$$

- ✓ Image generation from two images

$$\tilde{B} = m(A, B,\theta_m) \quad \text{…………………………………………………}(2.6)$$

- ✓ Image generation from image A and features of image B

$$\tilde{B} = m(A, f(B) ,\theta_m) \quad \text{………………………………………………}(2.7)$$

- ✓ Image generation from features of two images. This could happen if the model cannot handle intended feature extraction or detection or extraction of features are needed before training or before the data is fed to the model purposely. Disentangling Multiple Conditional Inputs in GANs (Gökhan Yildirim C. S., 2018), in which the generator model takes 3 inputs shape feature as ***f(A),*** texture feature as ***f(B)*** and color feature ***z*** , can be the best example since feature extraction is done as data preprocessing purposely to see the effect of each input on image to be generated using these inputs.

$$\tilde{B} = m(f(A), f(B),\theta_m) \quad \text{…………………………………………}(2.9)$$

$$\tilde{B} = m(f(A), f(B), z\ \theta_m) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.10)$$

Loss calculation techniques of Generative Adversarial Networks also follow similar procedure like during generation process. To demonstrate this let $\boldsymbol{y}$be reference image**,** $\bar{\boldsymbol{y}}$be predicted or generated image and d be distance function, loss calculation can be expressed as the following equations.

$$\ell = \begin{cases} d(y, \bar{y}) & for\ instance\ level\ loss \\ d\big(f(y), f(\bar{y})\big)\ for\ feature\ based\ loss \end{cases} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.11)$$

Where ℓ represents loss and f stands for feature extraction function.

Incase reference images are more than one equations(2.2,2.3,2.4,2.5,2.6,2.7,2.8,2.9,2.10), feature treatment is the crucial thing.

In Pix2Pix, sketch and random latent vector is used to generate image. the sketch feature is responsible for local structure loss and the extra information is used to control color and other texture information of generated image. but during loss calculation local loss for generator is mean absolute error or L1 which measures the distance between generated image and target image. but the reference image is RGB color image which means they confusing the model generalization of latent input as intended to make color difference. Because of this color of generated image is generalized from sketch information and latent vector added.

In Disentangling Multiple Conditional Inputs in GANs (Gökhan Yildirim C. S., 2018), texture feature extracted from generated image and input texture feature of the second reference images are used to calculate difference between reference and generated image while they were trying to preserve color information added. The laplacian matting matrix they used to extract color information extract texture information containing pattern made by spatial arrangement and pattern made by color shade. It may be difficult for their model to identify which color information they want to preserve since both texture provided and additional color inputted containing color information and as a result their model could not handle complex texture structure image.

In some papers Gram matrix and pre-trained model like VGG-19 were used as content feature and style feature extraction methods. In most of our related works boundaries between features or characteristics of image to be preserved and to be generated by model, to be translated or generalized from additional or extra information inputted, or to be transferred from the second reference image or reference domain do not specified or identified clearly.

If feature(s) of images extracted using the same algorithm of are similar, we can say that those images are similar. Texture is among features of image, which is used to represent structural varieties and or color varieties. Structural texture is based on intensity variation, while color texture depends on color varieties. Color variation based texture does not contain information about individual color contained in the image. So it is hard to represent color information of image using texture since it is based on color variation and ignores nature of those colors.

We summarized how related works treated the features of images in both preprocessing stage and during loss function calculation in Table 2) Related Works Feature Treatment Summary

TABLE 2) RELATED WORKS FEATURE TREATMENT SUMMARY

No	**Paper**	**Features Name**	**Description**	**Extraction Method**	**Failures Caused**
	Disentangling Multiple Conditional Inputs in GANs (Gökhan Yildirim C. S., 2018)	Shape	⇨ To control area in which the article object should be in ⇨ White area as foreground and back area as background	GrabCut algorithm	⇨ Shape is not controlled ⇨ Average color of this image is equal with that of input color but the generated image contains different color ⇨ Could not handle when texture

		Texture	⇨ Intended to control the local structure of the generate image.	⇨ Extracted using Lablacian Matting Matrix	used as input is extracted from image of complex texture structure and multiple color.
	Pix2Pix (Phillip Isola, 2017)	Sketch	⇨ To control local structure of image to be generated ⇨ L1 or Mean Absolute error is used to measure the loss.	⇨ Edge extraction algorithm	Color of ground truth image and generated image similarity is there, which means their model is enforced to generalize color from sketch and the extra information added.
		Random latent vector	⇨ To control diversity	⇨ generated randomly to be used as input	

	FashionGAN	Style feature (fabric pattern)	⇨ To control style of generated image ⇨ Fabric pattern patch is used for input ⇨ Encoded to latent vector before fed to generator	⇨ Gram matrix from conv2_2, conv3_2 and conv4_2 layers of VGG-19.	⇨ Fails to handle complex and multicolored fabric pattern ⇨ Additional part is generated with image. ⇨ Diversity made by the random latent vector added is not effective.
		Content feature (sketch or grayscale image)	⇨ To control local structure of image ⇨ Sketch and grayscale image are used as input	⇨ Sketch extraction techniques	
		Random latent vector	⇨ To control diversity of generated image	⇨ Random number generator	

2.3.5 Summary and Limitations of related works

FIGURE 7) GAPS OF FASHIONGAN AND DISENTANGLINGGAN

Figure 7) GAPS of FashionGAN and DisentanglingGAN Shows gaps of FashionGAN and DisentanglingGAN. Both works generates images using sketch and other extra information like texture, color and fabric patterns. But during the style image with complex texture and multicolored they cannot handle image generation. Especially in disentanglingGAN controlling is the objective or controlling the effect of input attribute on generated image but shape controlling technique they have test after their model training fails to control shape.

TABLE 3) SUMMARY OF RELATED WORKS

No	Paper	Goal of the Paper	Metrics Used	Gaps

1.	Pix2Pix	⇨ To develop General purpose model to solve image to image translation problems	⇨ Amazon Mechanical Turk (AMT) ⇨ AMT perceptual studies ⇨ FCN-score	⇨ Problem of handling multiple color ⇨ Problem of handling complex structure or texture
2.	TextureGAN	⇨ To develop deep image synthesis model guided by sketch, color, and texture	⇨ Human preference ⇨ Inception score	⇨ Problem of handling multiple color ⇨ Problem of handling complex structure or texture
3.	FashionGAN	⇨ To develop cGAN model for end to end virtual garment display	⇨ Human preference ⇨ Inception score	⇨ Problem of handling multiple color ⇨ Problem of handling complex structure or texture
4.	DisGAN	⇨ To disentangle the effects of different input features on image to be generated.	⇨ Log-likelihood ⇨ linear correlation	⇨ Problem of handling multiple color ⇨ Problem of handling complex structure or texture ⇨ Their model did not make interprets features clearly to what they are

				intended for.
5.	ClothGAN	⇨ To develop ClothGAN framework that automatically generate new patterns and styles of clothing with Dunhuang elements	⇨ Human preference score ⇨ Inception score (IS)	⇨ Problem of handling multiple color ⇨ Features of images are miss treated since the same feature extraction algorithm is used to extract style information and content information from different reference images

CHAPTER 3

3 METHODOLOGY

3.1 Overview of methodology

Under this chapter methodology used to solve the problem is discussed. First data oriented methodologies which are used specifically to deal with data are discussed and model oriented methodologies including baseline papers are discussed.

3.2 Specification of Required Data

Before collecting data there are things those should be specified based on nature and types of data needed for the model? These specifications are used to clearly defines the following three points

- ✓ What types of data going to be collected?
- ✓ Where to collect them?
- ✓ How to get them?

So the nature and or types of data going to be collected are specified by taking two major things in consideration.

- ✓ Model oriented: model takes some information as input, do some operations on input and generate new information. The objective of the model is to interpret the input to output. So then the input should provide all information needed for the interpretation process. Also during training process, the model needs information about what the output should looks like.
- ✓ User oriented: since the model is for users, to get output the user must provide the information needed by the model. The information should be not difficult to get or to provide to the model.

In our case the main objective of the model is to design fashion that reflects complex cultural patterns. When we say fashion it means a lot, fashion can be categorized into many categories or classes. Shoes, Pants, Shorts, Dresses, T-shirts, Bags, Hats and etc are some classes fashion or garment can be belonged under if the common attributes of features they share are concerned. Even if grouping all fashion images or garment items into specific group is difficult or probably

impossible, users need it. But we considered the computational requirements and user needs and we decided to select three classes of fashion namely Bags, Dresses and Shoes. So the model generates Bags, Dresses, and Shoes with cultural patterns.

By considering user needs and our model we decided the type and nature of data needed for our work. The following table shows them.

TABLE 4) SPECIFIED DATA PROPERTIES

No	Type of data	To be used		Type information intended to provide
		Input	Target	
	Type of Fashion(Bag, Dress, Shoes)			✓ Class or group level structural information
	Sketch or Edge Image	✓		✓ Item level local structural information
	Cultural Pattern Image	✓	✓	✓ Cultural pattern information
	Shape image	✓		✓ Shape information
	Target Fashion real image		✓	✓ Target image structural information

3.3 Data Collection Techniques

At user level, sketch and shape of fashion image can be drawn manually and does not require special techniques besides its simplicity. Since the type of image they want to be generated is important they simply feed the model with the name of type of image and the sketch and the area the target image should cover to the model. At model level local or global(group) level information can be provided in many ways. But most of them requires special techniques to get them besides their ability to make confusion during interpretation as they will carry another information.

For example, texture features can represent local structure of an image. but extraction process needs special techniques like Laplacian Matrix, GLCM (Gray Level Co-occurrence Matrix) and other color information will be included. By considering these things, we selected sketch and class name as sources of structural information and color patterns as color or style information sources for the model.

So then sketch, shape, class name and real target image can be extracted from single fashion image. Cultural pattern information which is to be used as input and target image can be provided as a single image. Now it is clear about the data going to be collected and we decided to collect these data for our model.

3.3.1 Dataset Construction

3.3.1.1 Fashion Dataset

Three classes namely Bags, Dresses and Shoes are categories or our fashion dataset. For shoes we used Pix2Pix dataset which contains 5,000 items of both sketch and target colored image stacked together. Dresses and Shoes are classes containing 5,000 colored images and they are collected from Generally, each items under each classes decided to contain sketch image, shape image and target image of single fashion image item. For shoes shape image is extracted through preprocessing, for Dresses and Bags both sketch and shape image are extracted by preprocessing phase of our work and attached together with their original image they are extracted from. Generally, our dataset contains 3 classes namely Bags, Dresses and Shoes. For each item in each classes 3 images (sketch, shape and real image) are included and as a result the number of images contained in our dataset are ***3 X 3 X 5,00 =47,000***.

3.3.1.2 Culture Dataset

This contains 2,000 images of many different cultures. First we collected more around 5,000 culture related images and stored into our data corpus folder. Then after among collected images we selected 2,000 images based on selection criterion we have specified. Image should contain multiple colors, pattern made by colors should be complex, percent of black and white color in an image should not be more than 30% and other legal and moral criterion were our selection criterions.

To collect these culture related images we have used online sources and local existing and available traditional cloths and or materials designer, traditional and cultural cloths or material resellers and stores. For online source we have used Pinterest images. Pinterest is Social Media designed to enable sharing and discovering of information on internet using images (Andrew Zhai, 2017). Also Pinterest is Image Sharing and Booking Website which is currently being described as Visual Search Engine (Hughes, 2013),(Andrew Zhai, 2017). We are not the first to use Pinterest dataset, Visual Discovery at Pinterest(Andrew Zhai, 2017) also used and tested on

visual search and recommendations services. The reason that we selected Pinterest is that it has been used as the place where people advertise their culture and because of that a lot of culture related images are included on their database. We have used Pinterest android mobile application that enables accessing images in their database. For local cultural patterns Keet traditional cloth and ornaments designer and distributor located at Adama town is used. But images we have collect from Keet requires a lot of preprocessing like segmenting fashion images and images contain a lot of black and white colors and because of these we have not included all of them in our dataset.

Generally, our culture dataset contains cultural patterns of different cultures like Ethiopian Cultural Patterns, African Cultural Patterns, Mosaic patterns, Mandala Patterns, Islamic Patterns and others

3.4 Design Methodologies

At earlier stage of model design phase components specification is used to specify all building blocks of the model, process identification is used to identify processes and activities of going to be performed for. Identification of relationship and interaction is used to identify the interaction between data and model and also to specify relationship among components of model.

Design of the model is blue print for model to be implemented. So at design phase everything should be represented and visualized correctly. Conceptual framework or representation of each processes and activities should be clearly represented and visualized. For this purpose, we have used the following things to make the design clear and simple.

- ✓ **Symbolical representation:** Features attributes going to be used like input features, output features and target features are represented using symbols. These symbols are to be used to visualized or used in mathematical equations used for further design. These symbols are used to identify and represents features.
- ✓ **Mathematical equation:** used to represent models, cost functions, optimizers, evaluation metrics and other algorithms used for design purpose.
- ✓ **Diagram or Graphs:** different shapes and likes are used to represent components and layers to represent graphically or by diagram.
- ✓ **Flow charts:** flow of data through model and between components and processes are to be represented using flow chart diagrams.
- ✓ **Pseudocode:** activities to be done by components or models using data are represented using Pseudocode which is usually used to represent implementation and or activities to be done.

3.5 Evaluation Techniques

Objective and subjective evaluation techniques are used to evaluate the performance of our model over unseen data or over test data.

3.5.1 Quantitative Evaluation

For quantitative evaluation of our model Inception Score is used. Inception Score is evaluation Metrics used to evaluate performance of model is terms of image quality generated by the model. Our baseline works, Pix2Pix and DisentangleGAN, also used this evaluation metrics to evaluate their model and we also choose Inception Score to help us to compare our work with baseline works.

3.5.2 Qualitative Evaluation

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

3.6 Feature extraction Techniques

Feature extraction processes are done at two different stages of our work. The first stage is during the dataset preparation and the second stage is before the model training. Since extraction of both sketch and shape images from 30,000 images may takes time and computationally needs a lot of memory and Processor we decided to extract them during dataset preparation. Sobel edge extraction is used for sketch or edge extraction. GrapCut is used for shape extraction method. But explanation of the extraction before model training is bulky and decided to be explained in succeeding three subsections.

3.6.1 Shape Mask Extraction

Shape feature of image is needed to be used as background and foreground separation feature. Binary image is used to represent shape feature. Background is represented by zeros (0) and foreground, which is used to show area of the image in which the target object is included, is represented by ones (1). GrabCut algorithm which is included as class in cv2 python package is

used to extract shape feature. Basically GrabCut is used to extract foreground and or background from image.

3.6.2 Grayscale Extraction

To extract grayscale image feature from original image and generated image mean of channels of each pixel is used. The following equation demonstrates how to calculate mean of channels in pixel at specific coordinate.

$$gray(x) = {\sum_{c=0}^{3} p_{i,j,c}}\Big/{3} \quad \text{... ... ...(3.4)}$$

3.6.3 Color Extraction

Where i=0,1,2,…,h and j=0,1,2,…,w and p is pixel of image x at (i,j) coordinate.

For color feature extraction we have developed new color feature extraction technique. The color feature extractor we have developed is ratio based color feature extraction. First sum of each channels in a pixel of an image, RGB, is calculated. Then the ratio of each channels to the sum is calculated. We have decided to develop this color feature extraction method because of that methods used for color feature extraction mechanisms used by previous works are believed to the one that exposed them for problems of handling multiple color with complex texture.

Let x represent image of w width and h height from where color feature is needed to be extracted from, and p represents pixel of the image. The following equation demonstrates the color extraction technique we have developed.

$$f(p_{i,j}) = {p_{i,j}}\Big/{\sum_{k=0}^{3} p_{i,j,k}} \text{ ..... (3.1)}$$

Where f is color feature extractor, $\boldsymbol{p_{i,j}}$ **is** is pixel of image at (i,j) coordinate and $\boldsymbol{p_{i,j,k}}$ **is** channel of pixel at (i,j) coordinate.

$$color(x) = \begin{bmatrix} f(p_{i=0,j=0}) & \cdots & f(p_{i=0,j=w}) \\ \vdots & \ddots & \vdots \\ f(p_{i=h,j=0}) & \cdots & f(p_{i=h,j=w}) \end{bmatrix} \text{ ..... (3.2)}$$

Where $\boldsymbol{color}(\boldsymbol{x})$is used to represent color feature vector of image $\boldsymbol{x}$

3.6.4 Sketch Extraction

Sobel-Edge extraction is used to extract edges of fashion image those are used as local structure controlling information or condition

3.7 Baseline

This section demonstrates previous works used as baseline for our work. These papers or works are those we used fully or partially for the development of our work.

TABLE 5) TABLE OF SUMMARY OF BASELINE WORK

No	Name	Year	Description
	FashionGAN	2018	✓ FashionGAN is the model developed to be general purpose for image to image translation ✓ Sketch to realistic photo is among tasks solved using FashionGAN ✓ Since the model we have proposed is similar in image generation from sketch and patterns we decided to use FashionGAN as baseline

3.8 Tools

3.8.1 Data Collection Tools

The following table shows the summary of tools we have used to collect and preprocess our data.

TABLE 6) DATA COLLECTION AND PROCESSING TOOLS

No	Tool Name	Type	Properties		Description
			Name	Value	
	Laptop PC	Hardware	Model	Toshiba Satellite C8	Since data collection and preprocessing is done before training we have decided to use this available resource
			RAM	4 GB	
			CPU	2.30 GHz	
			HDD	500 GB	
			Operating System	Windows 10	
	Camera	Hardware	Resolution		

3.8.2 Design Tools

The following table shows tools we have used to design our model and documentation process of our model design.

TABLE 7) DESIGN TOOLS

No	Tools Name	Tools Type	Properties		Description
			Name	Value	
1.	Edraw Max	Software	Version	7.7	✓ Used to assist drawing of diagrams
2.	Microsoft Office Word	Software	Version	2016	✓ Used for organization and documentation of the design ✓ Provides document conversion tasks like to word to pdf besides other uses
3.	Computer	Hardware	RAM	4 GB	✓ Edraw Max and Ms Office work are installed on it ✓ Used as storage and processing device
			CPU	2.30 GHz	
			HDD	500 GB	

CHAPTER 4

4 PROPOSED SYSTEM DESIGN AND ARCHITECTURE

4.1 Overview of Proposed System Design and Architecture

Under this chapter, proposed system is designed and presented. We divided into high level and low level design. Under high level the overall architecture including the main components relationship and interactions is designed and each of components are design under low-level subtitle.

4.2 Conceptual Framework

The proposed deep learning model is Generative Adversarial Networks (GAN) which have two model called Generator and Discriminator. The objective of our Generator model is to generate images with intended features, those can be classified as real images and in which class they are belonged can be correctly guessed by the Discriminator model. While the objective of our Discriminator model is to classify images from dataset as real and generated images as fake in addition to guess the correct class those images are belonged. Even though, these two models compete with each other in adversarial way to meet their goal, the very important thing is our target which is a point at which Nash-Equilibrium could be met.

To meet the primary goal of the work both models should be built from components those enables them to do the tasks they are intended. The architecture of these building components really matters and should be suitable for both models. Also data handling processes and provision environment and methods are very important. Finally training and evaluation mechanisms and methods are the crucial things towards the goal. So in this chapter they way data should be handled, the ways in which the models should be designed and the way training and evaluation should be done are going to be explained.

Let $\boldsymbol{d}$ represents the dataset, $\boldsymbol{g}(\ )$ represents the generator model, $\boldsymbol{d}(\ )$ represents discriminator model , $\boldsymbol{x}$ and $\boldsymbol{y}$ represent input variable and target variable respectively and$\bar{\boldsymbol{y}}$represent predicted and or generated output. But what should be clear is that input variables, target variables and predicted variables are different for both models. So we treated they as a set of and they will have different elements for each models. We demonstrated what they represent in each models in the following table.

TABLE 8) TABLE OF FEATURE REPRESENTATIONS

No	Symbols	Stands for	For Generator($g(\ )$)	For Discriminator($d(\ )$)
1.	x	Inputs features	$\{x_{label}, x_{sketch}, x_{pattern}\}$	$\{x_{sketch}, x_{content}\}$
2.	y	Target features	$\{y_{content}, y_{style}\}$	$\{y_{critic}, y_{label}\}$
3.	$\bar{y}$	Predicted features	$\{\bar{y}_{content\ +style}\}$	$\{\bar{y}_{critic}, \bar{y}_{lable}\}$

Based on their output their loss in calculated and according to their loss their trainable variables will be updated during training process. Let $\|\ \ \|$ represents the cost or loss function(s). like input features, output features and target features loss function could have different meaning. Since we will have multiple loss function we simplified by treating as distance symbol. Depending on the result of the loss function both generator and discriminator models update their trainable variables.

So we can say the crucial components we have to integrate are Data, Models both Generator and Discriminator, loss function and Optimizers to meet our objective. Before exploring into each component, designing integration of components is believed to be the first. So then the following diagram shows their integration including data flow through them.

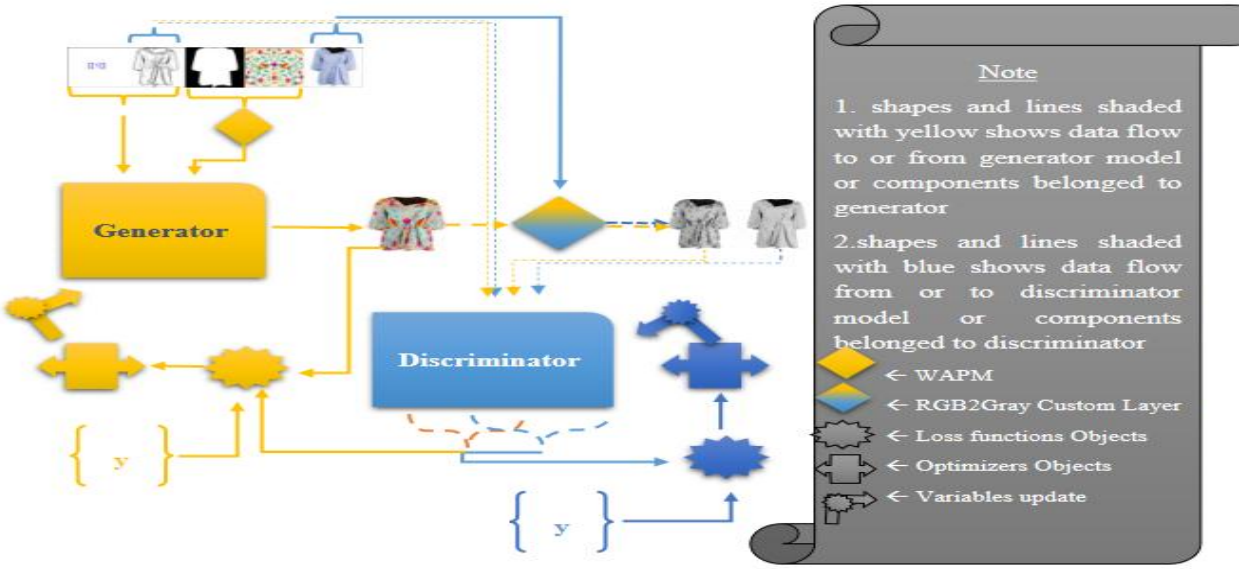


FIGURE 8) CONCEPTUAL FRAMEWORK

4.3 Data Handling Strategy

Under this subsection the way datasets should be arrange on storage and the way of preprocessing mechanisms will be designed and described. The following two subsections explore them.

4.3.1 Data Storage

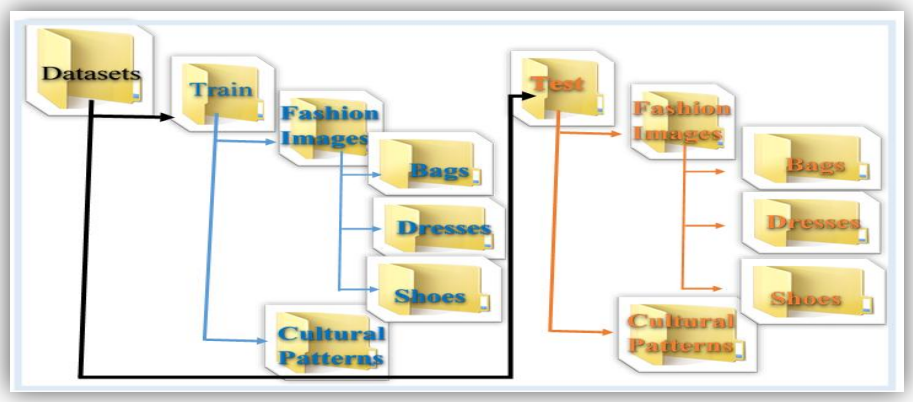


FIGURE 9) DATA STORAGE COMPONENT ARCHITECTURE

The Data Storage can be taken as one of building blocks of the system because of that it provides the data storing services for models. Both Generator and Discriminator models need data that is stored on this Data Storage component. This component is the place for the datasets. The way these datasets are to be arranged in the form that it could be easily accessed by other elements of the system using folders. The folder named with “Datasets” is the main folder for the data storage. This folder contains two subfolders namely “Train” and “Test” where two subfolders which are named “Fashion Images” and “Cultural Patterns” are belonged to each. In the “Fashion Images” folders, images classified into three classes namely Bags, Dresses and Shoes are included. In the “Cultural Patterns” folder, images of cultural patterns are included.

The general objective of this Data Storage component is to facilitate other components with training datasets and test datasets.

4.3.2 Data Pipeline

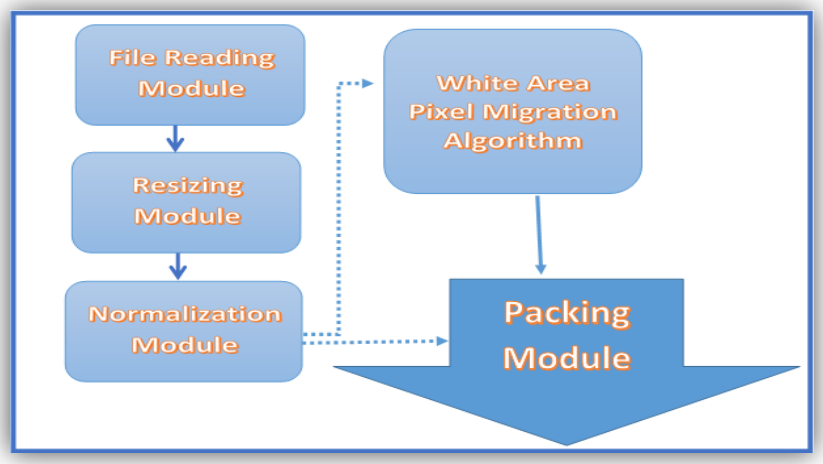


FIGURE 10) DATA PIPELINE COMPONENT ARCHITECTURE

The Data Pipeline component is the middle component between Data Storage and the models. The objective of this component is to read data from storage, performs some preprocessing activity and feed the model with these data. The Data Pipeline component contains five modules. The Input Reading Module is responsible for reading images from storage and outputs images with their perspective file names. For fashion images, target image, sketch of the target image and shape of the target image is stacked as a single image of size 256 x 768 x 3 which is to be

split into three images of 256 x256 x1. The resizing module is responsible for resizing images from input module with specified width and height. The Normalization Module is responsible for scaling intensity values to specified scale.

The very important part is the White Area Pixel Migration algorithm which is to be the custom assisting algorithm specified in the title of the paper. This module takes Cultural Pattern image, copy of target image and shape of target image. The shape image is to gives an information of the allowed area where pixels of pattern images are allowed to be in. by using this information the algorithm called White Area Pixel Migration Algorithm extracts representations of all pattern image pixels according to their ratio with the white pixels of shape image. this algorithm is like object reshaping but not as image resizing.

Algorithm 1: White-Area Pixel Migration algorithm. The custom assisting algorithm proposed and implemented to assist the deep learning model developed for our work.

Require: p_1, p_2 for storing transformed pattern images

Require: img_{shape} , $img_{pattern}$

Require: $f(img_{shape})$, function to count number of rows and columns containing at least one white pixel.

Require: $resize(x,(h,w,c))$ resizing function that resize x into h rows, w columns and c channels

Require: $ones\_containing\_ones\_count(img_{shape})$ is function counts number of ones contained in a ones containing rows.

Require: $d(p_2, v, img_{shape})$ function that assign v to the pixels of p p_2 at corresponding coordinates of img_{shape} white pixels.

h,w$\leftarrow f(img_{shape})$

$p_1 \leftarrow resize(img_{pattern}, (h,w,3))$

$One_{rows} \leftarrow ones\_containing\_ones\_count(img_{shape})$

For i in one_{rows} :

$V_i \leftarrow resize(p_{1_i}, (one_{row\ i}, 3)$

$new_{pattern} < -d(p_2, v, img_{shape})$

The output of this algorithm is 256 x 256 x 3 sized image with pixels at white pixel coordinates of shape are from cultural patterns image pixels' representatives and all pixels at black pixel coordinates of shape image are pixels from target image. The Packing Module is responsible for packing inputs as X and y features for the models. Class IDs are extracted here.

Finally, the output data from data pipeline properties of data are summarize in the following table.

TABLE 9) TABLE OF OUTPUT DATA FROM DATA PIPELINE

No	Features	Name	Data type	Shape	Scale
1	Input features	Class label	Integer	(1,)	[0,2]
		Sketch image	Float 32	(256,256,1)	[0,1]
		Pattern image	Float 32	(256,256,3)	[0,1]
2	Target features	Class label	Integer	(1,)	[0,1]
		Real image	Float 32	(256,256,1)	[0,1]
		Pattern image	Float 32	(256,256,3)	[0,1]

4.4 The Generator Model Architecture

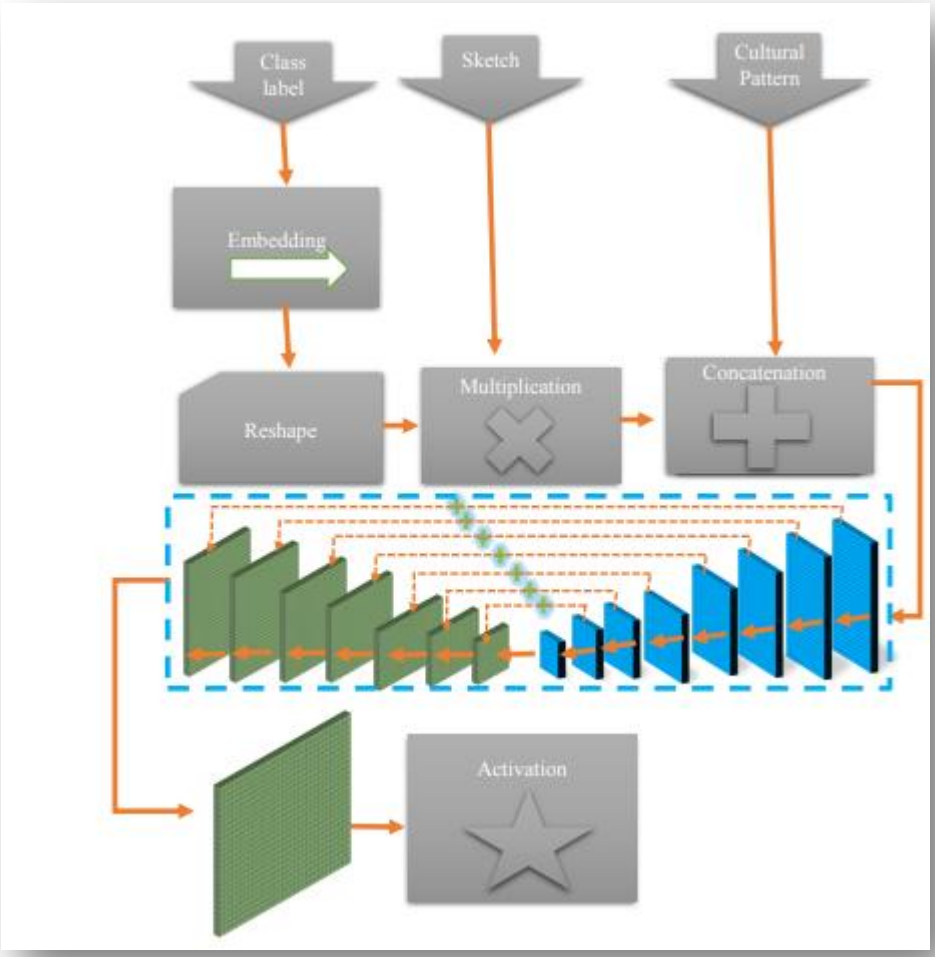


FIGURE 11) GENERATOR MODEL ARCHITECTURE

Architecture of our generator model is U-Net which is used. But some modifications are made by adding three layers before the U-Net and one up sampling block after the U-Net to the generator. The three layers added before the U-Net, are Embedding layer, Multiplication layer and Concatenation layer. These three layers are intended to transform the three inputs submitted to the generator into a single instance before they fed to U-Net. The Embedding layer takes the class label of shape (1,) and integer data type as input and transforms it into output of shape (256,256,1) tensor type, which is going to be multiplied with sketch input of shape (256,256,1) by the Multiply layer which outputs tensor of shape (256,256,1). After, the Concatenation layer concatenates the output from Multiplication layer and color pattern input image of shape

(256,256,3) together and produces tensor of shape (256,256,4) which is direct input to the U-Net block. The U-Net block consists eight (8) down sampling blocks, eight (8) up sampling blocks and seven (7) Concatenation Layers which are used as linking between down sampling blocks and up sampling blocks. Generally, the U-Net part of the generator can be categorized into two groups namely encoder block and decoder block. The encoder block is series of up sampling blocks and the decoder block is series of down sampling blocks.

All down sampling in encoder block is shares many characters with each other and differs on few things. Like encoder block, all down sampling blocks of the encoder blocks share many features with each other. Sharing many characters possibly leads to duplication of line of code during implementation which in turn causes implementation complexity. So we designed the way by which this implementation complexity can be reduced. We design function that contains similar layers they have in common and accepts modification could be made. The function returns object of Sequential class having common layers and layers belongs to individual.

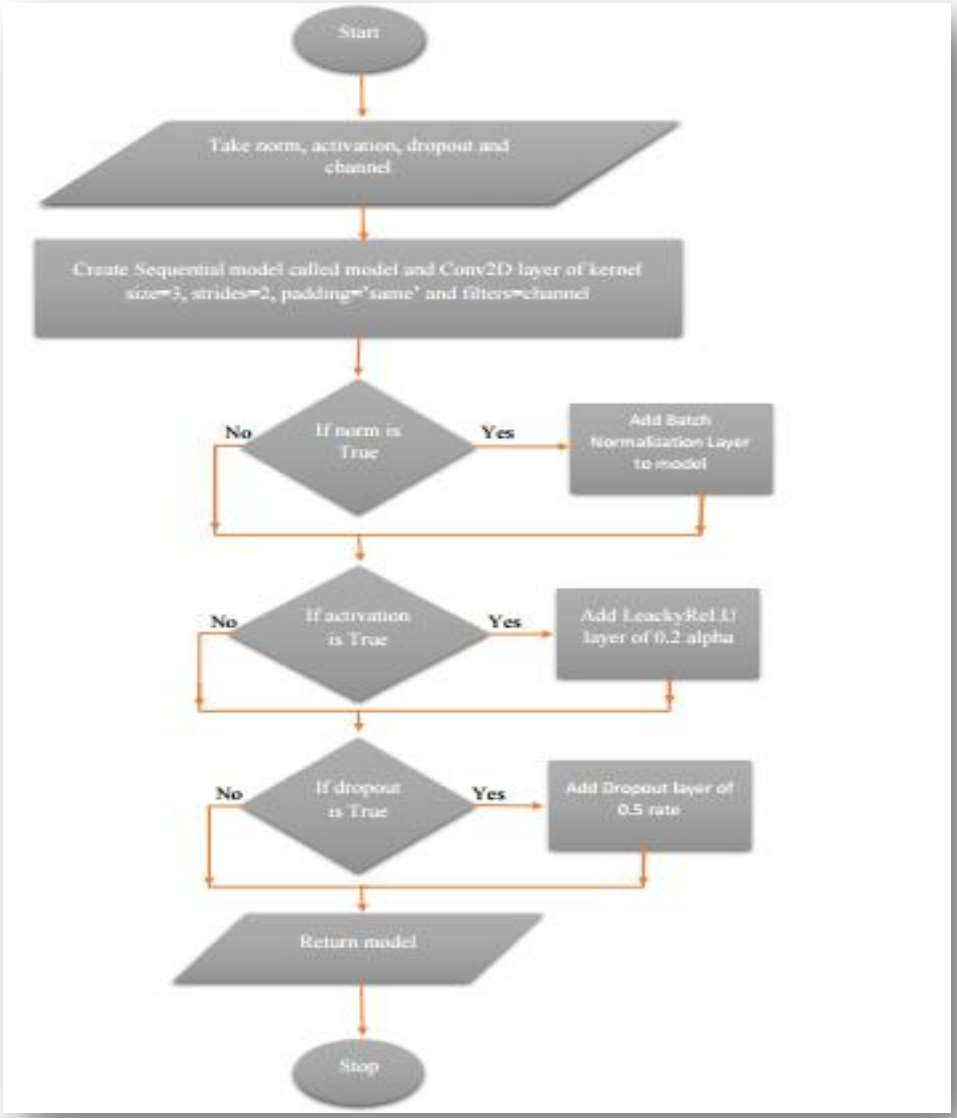


FIGURE 12) ENCODER OBJECT CREATION FLOW CHART

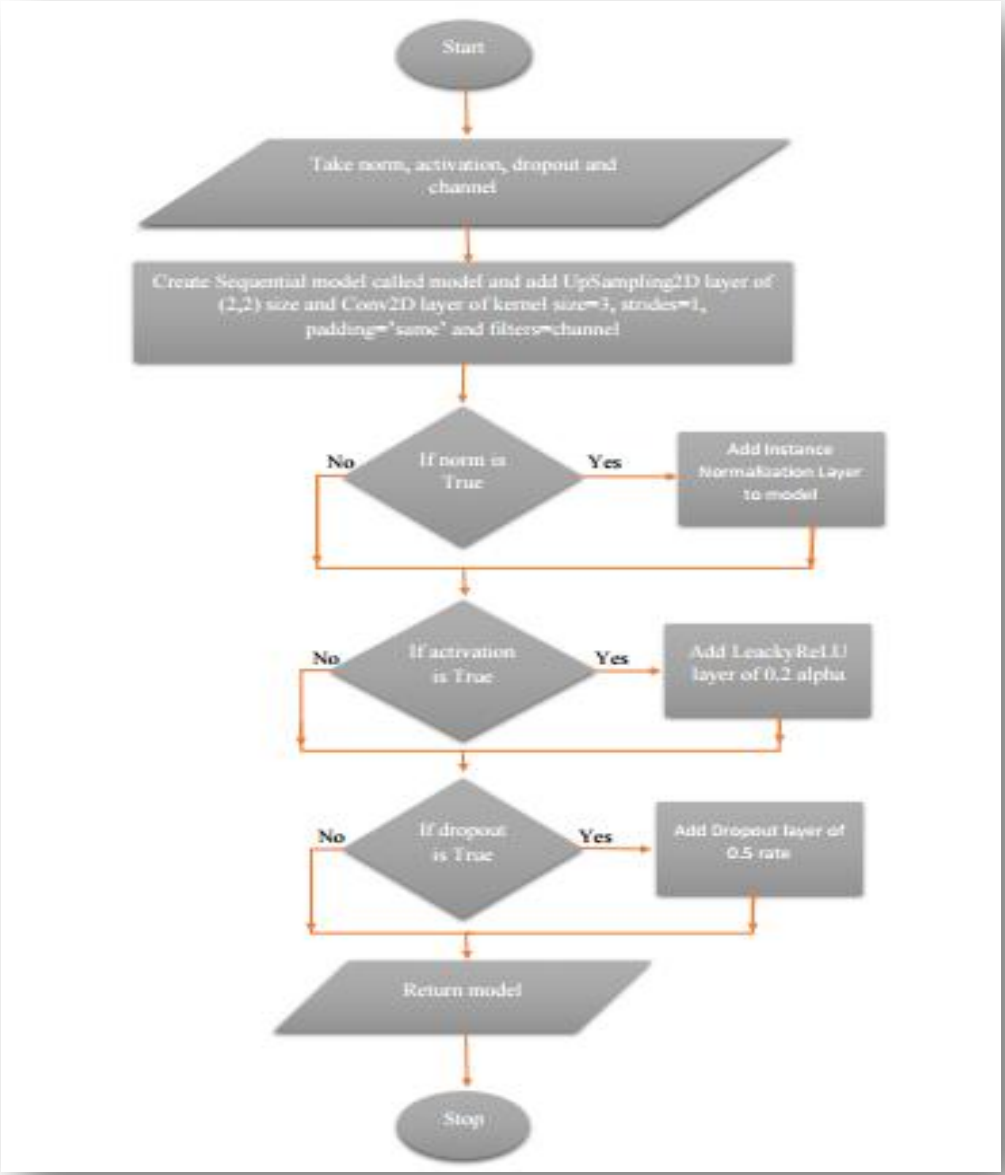


FIGURE 13) DECODER OBJECT CREATION FLOW CHART

A. The Encoder

The normalization layer, activation layer and dropout layers are layers by which down sampling object can be different and also their output channel can be different. But they are similar by kernel size=3, strides=2, padding='same' and one 2DConvolutional layer with these common attributes. The function returns object of Sequential class. The function need information whether or not to add normalization layer, activation layer and dropout layers to the object

requested to be created. Even though addition of these layers to the object to be return is optional the nature of these layers and their attributes are predefined, the normalization layer is Batch Normalization; the activation layer is LeackyReLU of 0.2 *alpha* and Dropout layer of 0.5 *rate*.

To create each up sampling block we pass output channel which is integer, three Boolean values to decide inclusion of normalization layer which is BatchNormalization, activation layer which is LeackyReLU and droupout layer which is Dropout of 0.5 rates respectively. The encoder is treated as list of down sampling objects. The following table summarizes the design encoder blocks with their properties.

TABLE 10) TABLE OF ENCODER OBJECT PROPERTIES

Down Sample at index	Normalization	Activation	Dropout	Output Channel	Input Shape	Output Shape
1	False	True	False	64	(256,256,4)	(128,128,64)
2	True	True	False	128	(128,128,64)	(64,64,128)
3	True	True	False	256	(64,64,128)	(32,32,256)
4	True	True	False	512	(32,32,256)	(16,16,512)
5	True	True	False	512	(16,16,512)	(8,8,512)
6	True	True	False	512	(8,8,512)	(4,4,512)
7	True	True	False	512	(4,4,512)	(2,2,512)
8	True	True	False	512	(2,2,512)	(1,1,512)

B. The Decoder

For up sampling in decoder block normalization layer is InstanceNormalization, activation layer is LeackyReLU of 0.2 alpha, dropout layer is Dropout of 0.5 rate and they are optional which can be decided at their object creation. Additionally, Upsampling layer of size (2,2) and Convolutional layer with optional output channel, kernel size ‘same’ and strides of size 1 are added. The mechanism of controlling optional features is designed and shown in the following flow chart.

The following Table 11) Table of Decoder Block Properties summarizes all up sampling blocks used in our U-Net model.

TABLE 11) TABLE OF DECODER BLOCK PROPERTIES

Up Sample at index	Normalization	Activation	Dropout	Output Channel	Input Shape	Output Shape
1	True	True	True	512	(1,1,512)	(2,2,512)
2	True	True	True	512	(2,2,1024)	(4,4,512)
3	True	True	True	512	(4,4,1024)	(8,8,512)
4	True	True	True	512	(8,8,1024)	(16,16,512)
5	True	True	True	256	(16,16,1024)	(32,32,256)
6	True	True	False	128	(32,32,512)	(64,64,128)
7	True	True	False	64	(64,64,256)	(128,128,64)

The U-Net part of our generator contains encoder block, decoder block and the skip connection between encoder and decoder. The link connection is Concatenation layers which are seven in number. Output from the U-Net part of our generator model has (128,128,128) shape. The output from the U-Net passes through another up sampling block which its output channel is 3, activation is None, kernel size is 3, strides size is 1, normalization is None and no dropout. The reason we made the activation of this up sampling block is that the predefined activation function is LeackyReLU which is not to be used at output of our generator. So we decided to use Tanh activation function to pass the last output or the intended image to be generated through. So the Tanh activation layer is the last layer of our generator model. In the following table we summarized our generators' building block by treating the U-Net part as single part.

TABLE 12) TABLE OF U-NET COMPONENTS AND LAYERS PROPERTIES

No	Layer or layers block Name	Direct Preceding Layer or layers block	Input shape	Output shape
1.	Embedding Layer	Input layer for class label	(1,)	(256,256,1)
2.	Multiply Layer	Embedding layer Input layer for sketch image	(256,256,1) (256,256,1)	(256,256,1)
3.	Concatenation Layer	Multiplication layer Input layer for color pattern	(256,256,1) (256,256,3)	(256,256,4)
4.	U-Net Part	Concatenation layer	(256,256,4)	(128,128,128)
5.	Up sampling Block	U-Net Part	(128,128,128)	(256,256,3)
6.	Tanh activation layer	Up sampling Block	(256,256,3)	(256,256,3)

4.5 The Discriminator Model Architecture

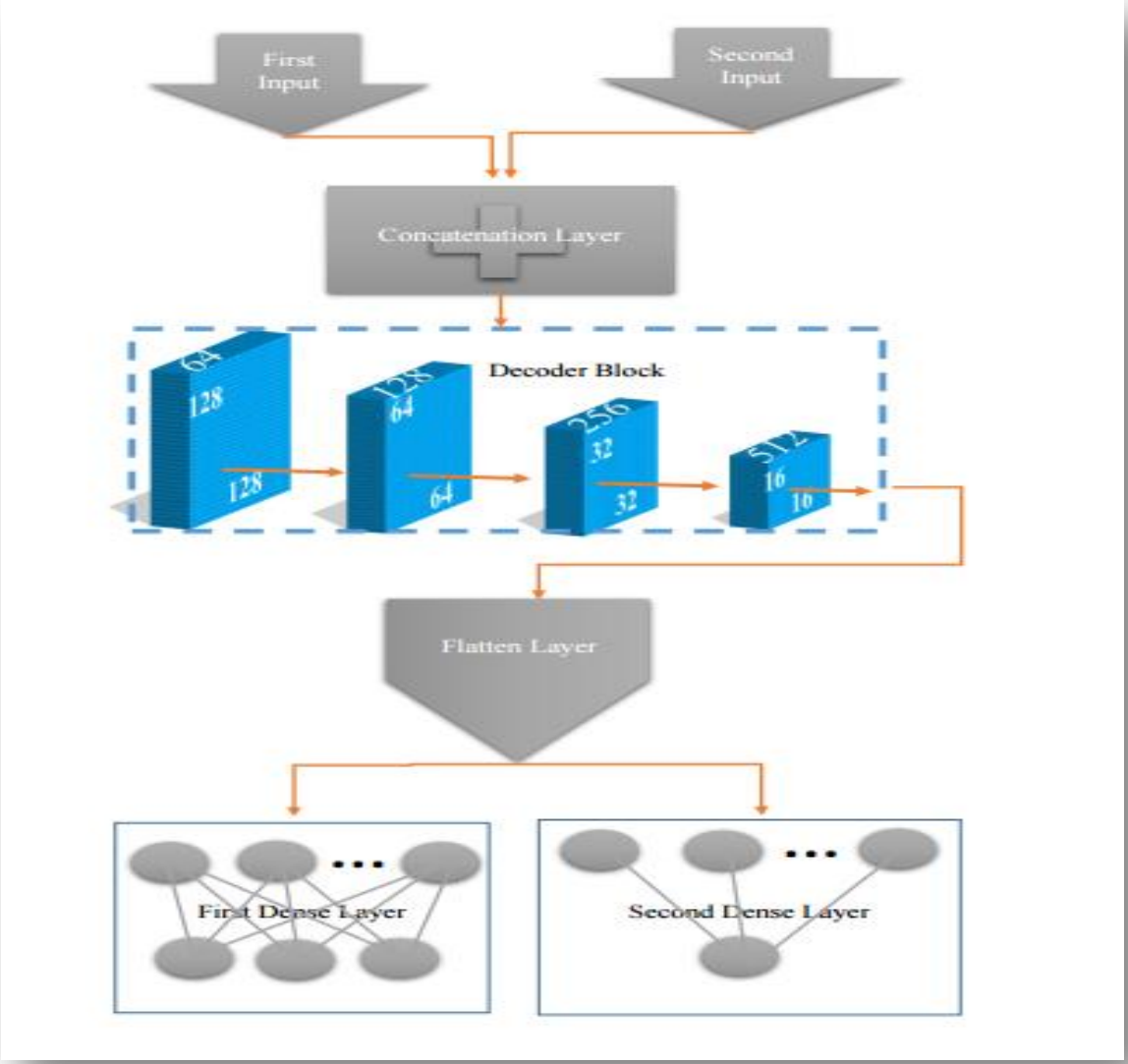


FIGURE 14) DISCRIMINATOR MODEL ARCHITECTURE

4.6 Training Strategy

Our generator model and discriminator model training is simultaneous or parallel. First generator generates fake images and using generated fake image and corresponding real image the discriminator model trained and then after generator model training follows. This iteration continues until the required object is met or until our models weights cannot be updated or modified which may be caused by quality of dataset or conflicts between loss functions.

4.7 Loss Calculation Techniques

- ✓ **Local Structure loss**: we used local structure loss to enforce the generator to translate local structure of intended image from input sketch image. L1 is used here like in Pix2Pix but here L1 is calculated between feature of generated image and target image.

$$\mathcal{L}_{ls} = \left\| G(\hat{e}_y) - e_y \right\|_1 \quad \text{(4.1)}$$

- ✓ **Color Style Loss:** we used L2 between color feature of predicted image and real or target image.

$$\mathcal{L}_{cs} = \left\| G(\hat{e}_y) - e_y \right\|_2 \quad \text{(4.2)}$$

- ✓ **Adversarial Loss:** used to measure the probability of image from generated distribution or real distribution. It is a king of binary classification problem and we used Binary Cross entropy loss function.

$$\mathcal{L}_{Adv} = \frac{1}{output\ \_size} \sum_{i=1}^{output\ \ size} y_i \,.\log(\hat{y}_i) + log(1 - y_i).\, log(1 - \hat{y}_i) \quad \text{(4.3)}$$

$$\mathcal{L}_{adv}^{D} = \mathrm{E}_{x,y\ P_{data}\ (x,y)}\left[\log D(x^{image}\ , y^{image}\ )\right] + \mathrm{E}_{x\ P_{data}\ (x)}\{\log[1 - D(ximage,Gey)] \quad \text{(4.4)}$$

$$\mathcal{L}_{adv}^{G} = \mathrm{E}_{x\ P_{data}\ (x)}\{\log[1 - D(x^{image}\ , G(\widehat{e_y}))]\} \quad \text{(4.5)}$$

- ✓ **Auxiliary Loss:** since our discriminator model is employed for critics and predicting the most likely class of images we used Auxiliary Loss instead. Sparse categorical cross entropy is used for this loss function. We used Sparse Categorical Cross Entropy Loss since our generator model is used to predict class of fashion items belonged. It means the problem here is multi-class classification problem.

$$\mathcal{L}_{Aux} = -\sum_{i=1}^{output\ \ size} y_i \,.\log(\hat{y}_i) \quad \text{(4.6)}$$

$$\mathcal{L}_{Aux}^{D} = \mathrm{E}_{x\ P_{data}\ (x)}\left[-\sum_{i=1}^{output\ \ size} y_i \,.\log(D(x)\right] \quad \text{(4.7)}$$

$$\mathcal{L}_{Aux}^{G} = \mathrm{E}_{x\ P_{data}\ (x)}\left[-\sum_{i=1}^{output\ \ size} y_i \,.\log(D(G(x)\right] \quad \text{(4.8)}$$

- ✓ **Total Loss**

Based on the above four formulas we calculated loss of our generator model and discriminator model according to the followings.

$$\mathcal{L} = \min_G \max_D [\lambda_{ls} \boldsymbol{\mathcal{L}_{ls}} + \boldsymbol{\lambda_{ls}\mathcal{L}_{cs}} + \boldsymbol{\lambda_{Aux}}\mathcal{L}^G_{Aux} + \lambda_{adv}\mathcal{L}^G_{adv}] - [\lambda^D_{Adv}\mathcal{L}^D_{adv} + \lambda^D_{Aux}\mathcal{L}^D_{Aux}] \ldots. (4.9)$$

4.8 Optimizer and Optimizer's Parameters Selection Techniques

For both of our models, Generator and Discriminator model ADAM (Diederik P. Kingma, 2017) with learning rate of 2e-4, β_1 of 0.5 and β_2 of 0.99 is used.

4.9 Data Train Test Split Techniques

We split our dataset based on recommendation of Effect of Dataset Size and Train/Test Split Ratios in QSAR/QSPR Multiclass Classification (Anita Rácz, 2021), using 70% or 80% splitting ratio of training and testing dataset is best of large dataset. We used 75% of our dataset which is 11,250 for training and 25% of our dataset which is 3,750 for testing purpose. We have chosen this ratio, 75%, which is average of 70% and 80% to get advantage of both.

4.10 Evaluation Mechanism

Even though fashion image generation is our primary objective, other supportive tasks, classifying into fake and real classes and predicting classes in which the fashion item is belonged or should be belonged, should be performed. So to meet our objective both of our models, generator and discriminator models, are evaluated besides evaluation of generator model in terms of generated images quality and diversity. For classification into fake and real or binary classification performance Binary Cross Entropy Accuracy is used, for prediction of fashion items classes Sparse Categorical Cross Entropy Accuracy is used to measure performance of our model during training parallel with training. But evaluation of our models on test dataset is not parallel or simultaneous of training because of loads may be imposed on training and test accuracy is measured separately after training. Evaluation of generator using Inception Score is done on generated images is done.

CHAPTER 5

5 IMPLEMENTATION

5.1 Overview of Implementation

Under this chapter implementation of the system is discussed. Source code is screenshotted and attached in appendices and other information about implementation is discussed and included in the body of this chapter.

5.2 Environment Setup

- ✓ Dataset: Fashion Dataset and Culture Dataset
- ✓ Programming Language: Python
- ✓ Computing Framework: TensoFlow 2.9 and Keras
- ✓ Computing Hardware: Dell Desktop Computer of 3.7 GHz CPU and 4 GB RAM and GPU of 12 GB RAM
- ✓ Other supporting Python Packages: cv2, Numpy, Matplotlib, and Skimage

Table 13) Environment Setup shows properties of some packages and or frameworks used.

TABLE 13) ENVIRONMENT SETUP

No	Tools Name	Type of Tools	Properties		Description
			Name	Value	
1	Tensorflow		Version	2.9	Deep learning framework
2	Anaconda	IDE	Version	3.0	Packing python and jupyter notebook
3	Jupyter Note Book	Editor	Version	2.9	For code editing purpose
4	Python	Programming language	Version	2.9	Compiling purpose and code management
5	OpenCV	Python Package	Version	2.0	Feature extraction and other tasks

5.3 Data Pipeline Implementation

For data pipeline we have developed one class that is used for data generation. For this data generator class inherits Sequence class from Keras.utils and other preprocessing functions are included as class methods of this class. For the implementation source code refer Appendix 1.

5.3.1 White Area Pixel Migration Implementation

This is our Assisting Custom Algorithm. To implement we have used numpy and tensorflow packages. For the custom assisting algorithm source code please refer Appendix 2 and for color feature extraction implementation refer Appendix 3.

5.4 Layers and Blocks Implementation

5.4.1 Down Sampling Block

For down sampling we have developed function or method the returns object according to the parameter passed to it. This method or function is used to handle code complexity and redundancy. This bock is used by function used to define generator and discriminator. Refer Appendix 4 for source Code.

5.4.2 Up Sampling Block

Like for down sampling block, we have developed function or method, that takes parameters used to determine properties of the object to be created and returned, for up sampling. Table shows information about the method or function developed for up sampling. Refer Appendix 5 for source code.

5.4.3 Gray Scale Extractor Custom Layer

The gray scale extractor class is implemented to extract grayscale image from original image and generated image before they are fed to discriminator model. And also the output of this layer is used as inputs for content extractor module. The reason we have developed this layers is to that if the discriminator model takes generated image and original image it can simply understand their difference and classify all generated image as a fake since generated image is different from original reference image by color.

5.4.4 UNET Implementation

Using down sampling blocks and up sampling blocks we have implemented UNET which is the main part of our generator model. for implementation of our UNET we implemented function or methods the build model from down and up sampling blocks and return it. Figure 15) Plotted UNET Diagram plotted UNET and for implementation source code refer Appendix 6.

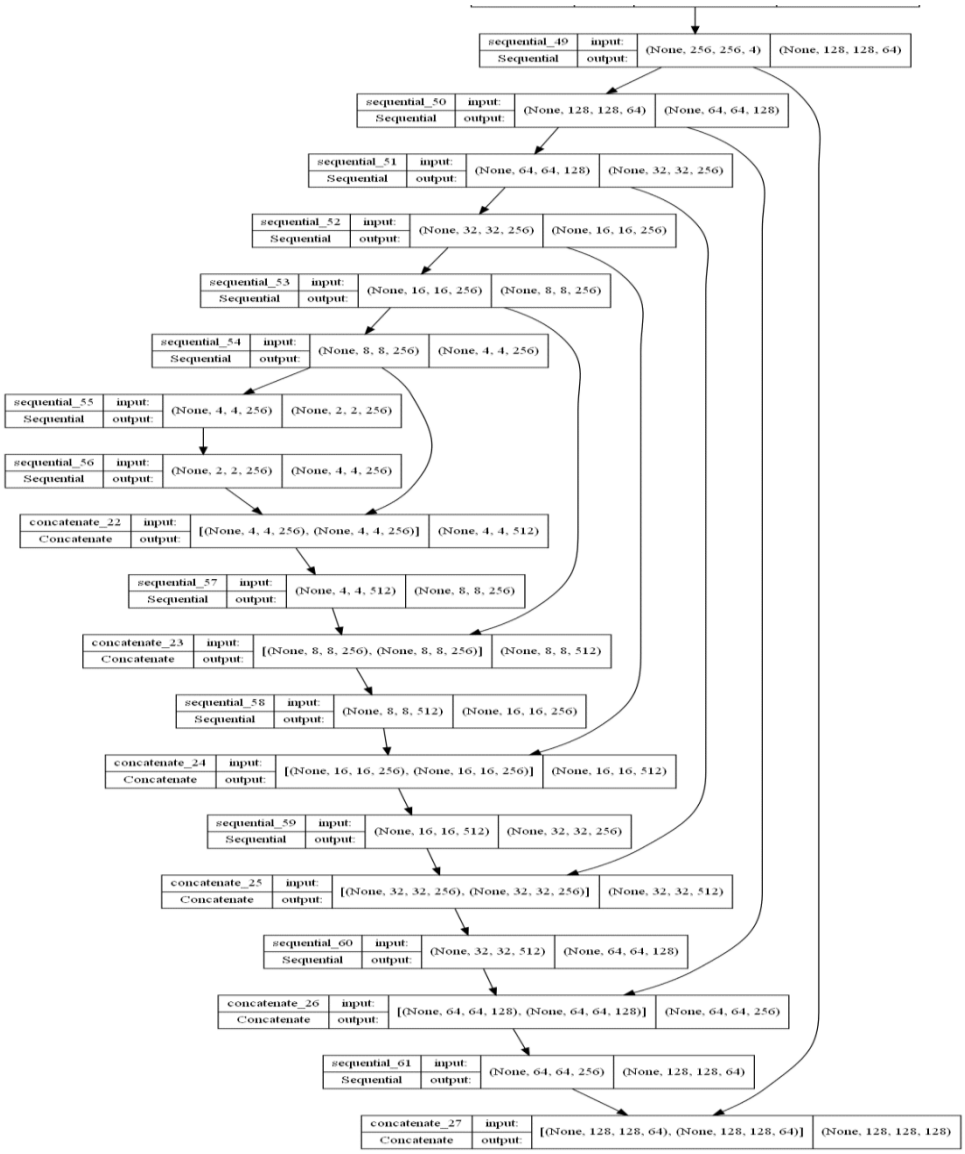


FIGURE 15) PLOTTED UNET DIAGRAM

5.5 Generator Model Implementation

We have implemented function or method that defines layers according to passed parameters and pre-specified parameters and return object of model object that represents our generator model. Using the API provided by TensorFlow we have shown summary generator model by figure. For source code refer Appendix 7

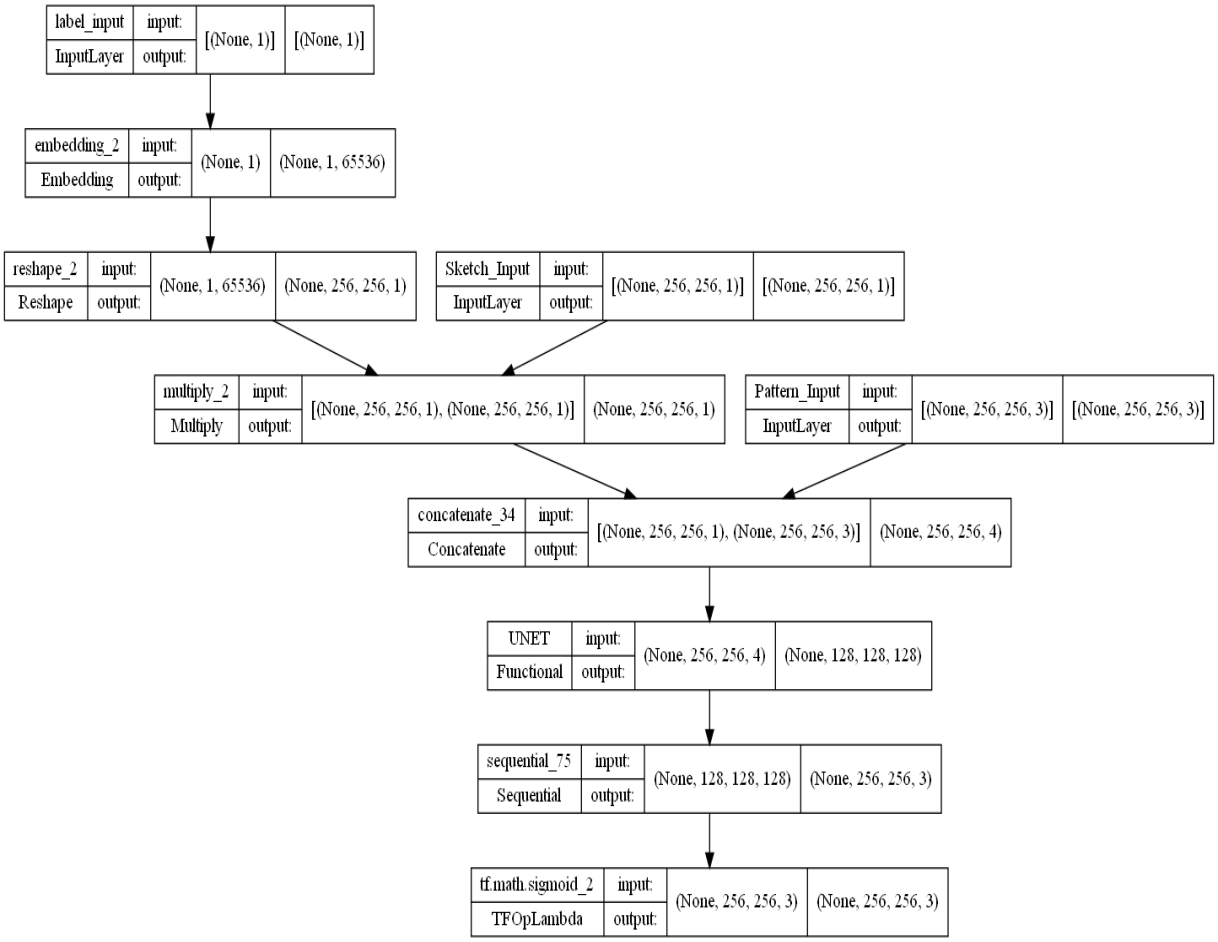


FIGURE 16) PLOTTED GENERATOR MODEL

5.6 Discriminator Model Implementation

Figure 17) Plot of Discriminator model shows the diagram plotted for Discriminator model during implementation and Appendix 8 shows its implementation.

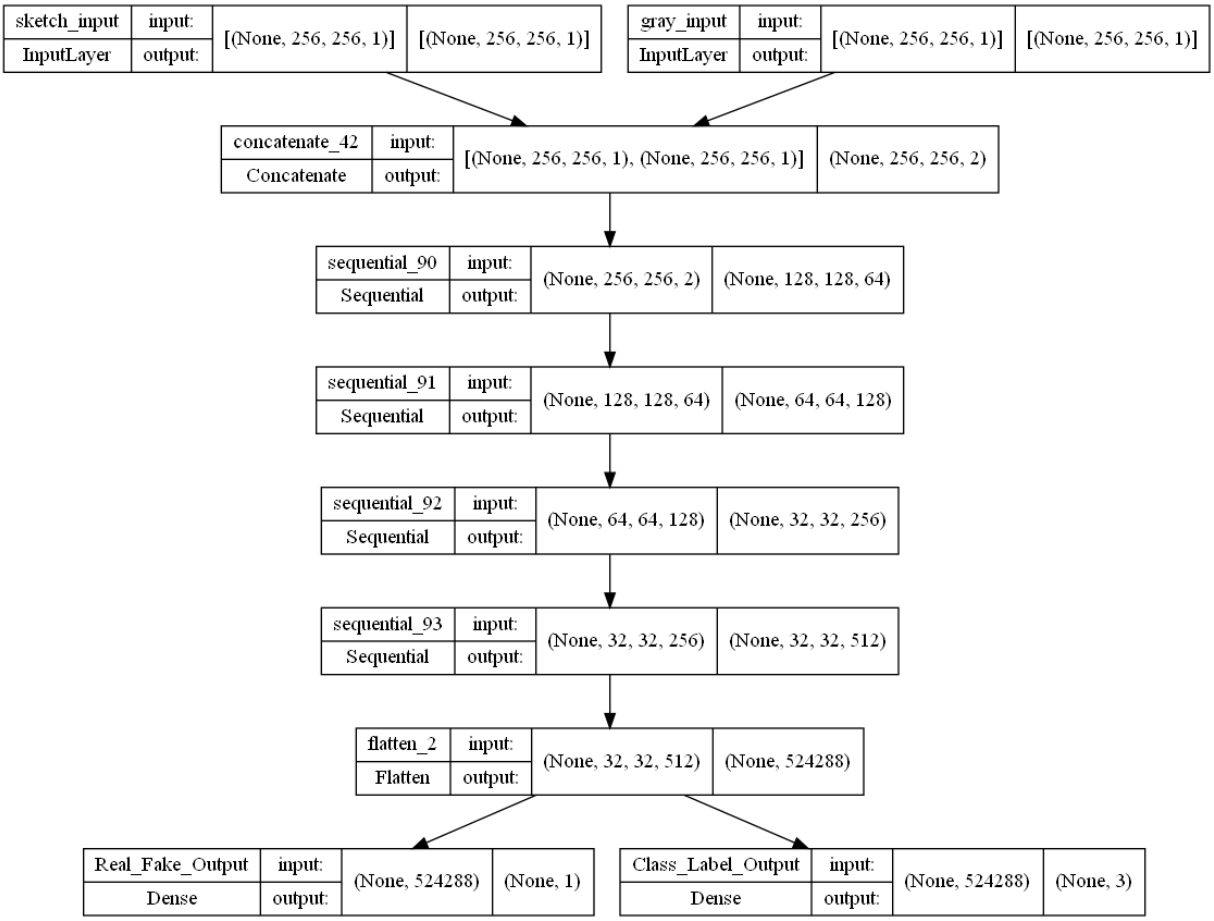


FIGURE 17) PLOT OF DISCRIMINATOR MODEL

CHAPTER 6

6 RESULTS AND DISCUSSIONS

6.1 Overview of Results and Discussions

In preceding chapters, ideas are coined, methodologies, algorithms and models are proposed, designed and implemented and this chapter discus and analysis results, evaluate our models and algorithms including results gained from them and present comparative study done on baseline works and our work.

6.2 Analysis and discussion of results of data handling procedure

Since deep learning is data oriented, this subtitle demonstrates how our dataset is handled and processed.

6.2.1 The Dataset Preparation Results

During dataset preparation some preprocessing mechanisms like edge extraction, shape mask extraction, resizing and pairing edge, shape mask and target image together are implemented and our dataset contains data passed through these preprocessing. Figure 18) Result from Data Pipeline illustrates how our dataset looks like.

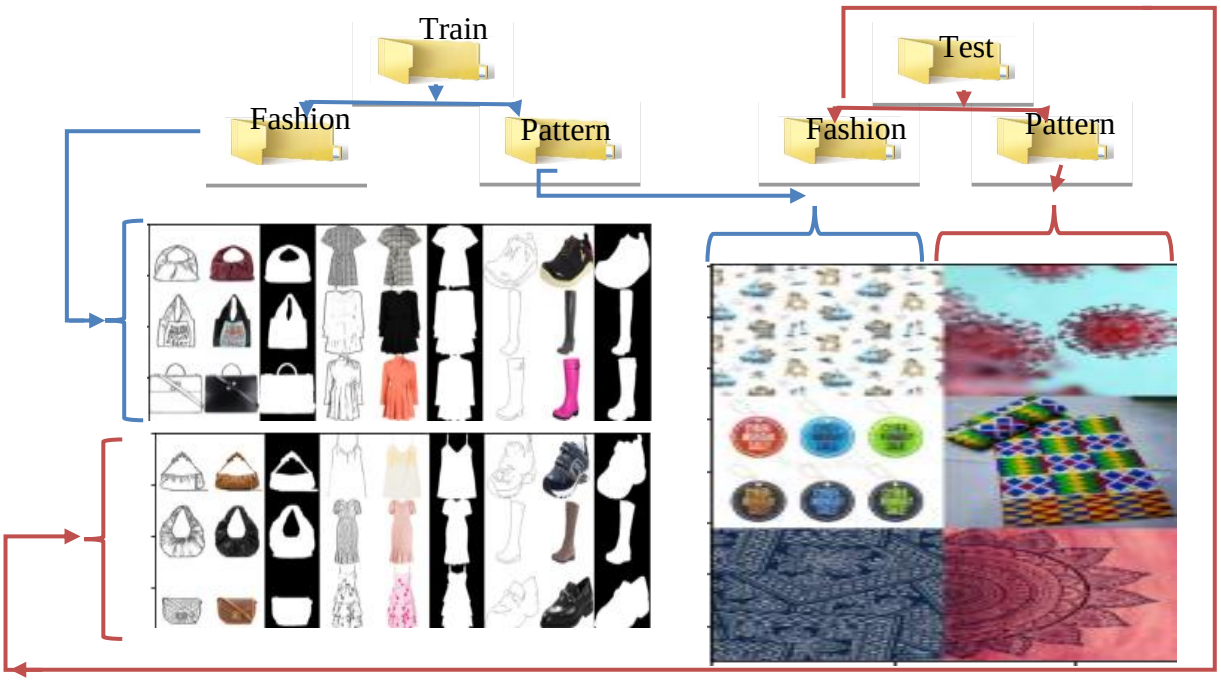


FIGURE 18) RESULT FROM DATA PIPELINE

6.2.2 The Custom Assisting Algorithm

Since the pattern input is to determine color feature of images of fashion objects, our target is on the pixels represent the object but not pixels representing background. So the custom assisting algorithm enables us to measure color similarities of pixels representing fashion object in the images and the pattern image.

FIGURE 19) RESULTS FROM THE ASSISTING ALGORITHM DEVELOPED

6.2.3 Evaluation of the Assisting Algorithm

We called it “White-Area Pixel Migration Algorithm”. The name of the algorithm explains the task of the algorithm. First the algorithm recognizes pixels of the shape mask with ones, which mean white area of the image as area of the object and zero (0) valued pixels as background of the object and then after enforces the whole pixels of pattern image to have representative in white area or compresses pixels of pattern image to be in corresponding mask white-area coordinates. Generally, it is a king of geometric transformation algorithm. A lot of questions can be raised on this algorithm.

For example:

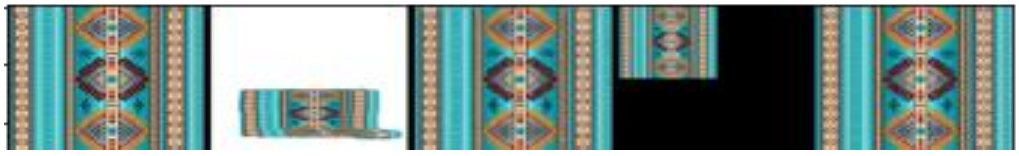
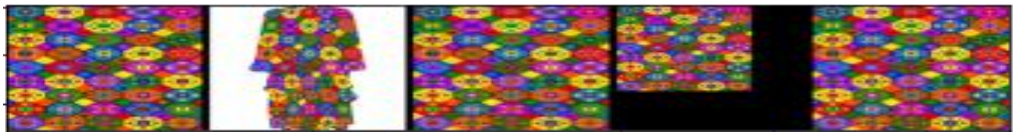
- ✓ Does the transformed image represent its original image?
- ✓ Does the transformed image contain all color information as the original image does?
- ✓ And others

Actually we are the first one to deal with these questions because the result can mislead our models which is dangerous for our objective. But the question is how? To answer the above questions, we did reverse engineering on the transformed image to get 3rd image having the same dimension with the original or the first image. the we measured similarities between the 3rd image and the first image by having in mind that losing too much information means the transformation procedure or the algorithm is not working otherwise it is working. For that reason, we decided to measure the performance of our algorithm with other algorithm having similar objective. So we selected bi-linear image resizing algorithm.

The procedure:

- ✓ For a given image x we applied transformation using our algorithm and Bilinear image resizing algorithm to get $\boldsymbol{a}$ representing image transformed by our algorithm and $\boldsymbol{b}$ representing image transformed by Bilinear image resizing algorithm.
- ✓ Reversing processes of both algorithms back to get a $\widehat{\boldsymbol{a}}$ and $\widehat{\boldsymbol{b}}$ representing images retransformed by our algorithm and Bilinear image resizing algorithm respectively
- ✓ Structural Similarity Index (SSIM) of $\widehat{\boldsymbol{a}}$and $\widehat{\boldsymbol{b}}$with x is calculated to see the difference and evaluate performance of our algorithm.

TABLE 14) ANALYSIS AND EVALUATION OF WHITE-AREA PIXEL MIGRATION

No		x	a	$\widehat{\boldsymbol{a}}$	b	$\widehat{\boldsymbol{b}}$
1	Images					
	SSIM	0.7044887			0.7228033	
2	Images					
	SSIM	0.9070184			0.8990245	

3	Images		
	SSIM	0.66614395	0.8382273
Average		0.759217	0.820018

As it is possible on table 14 we have calculated Image Similarity index between retransformed image from our transformation algorithm and bilinear image resizing algorithm. Using this method, we have calculated SSIM for 20 images using both algorithms and we got our algorithm only 0.0024 far from the second algorithm. For our algorithm we got 0.772877 and 0.775277 for the other. Even though there is a difference we calculating at objective function it is difficult to point out their difference from original image.

6.2.4 Color Feature Extraction Results

Color feature is extracted from fashion image after transformed by the assisting custom algorithm. Our color feature does not content features of the image it is extracted from. The color feature vector has the same dimension as the image it is extracted from. The following diagrams shows what color features extracted looks like.

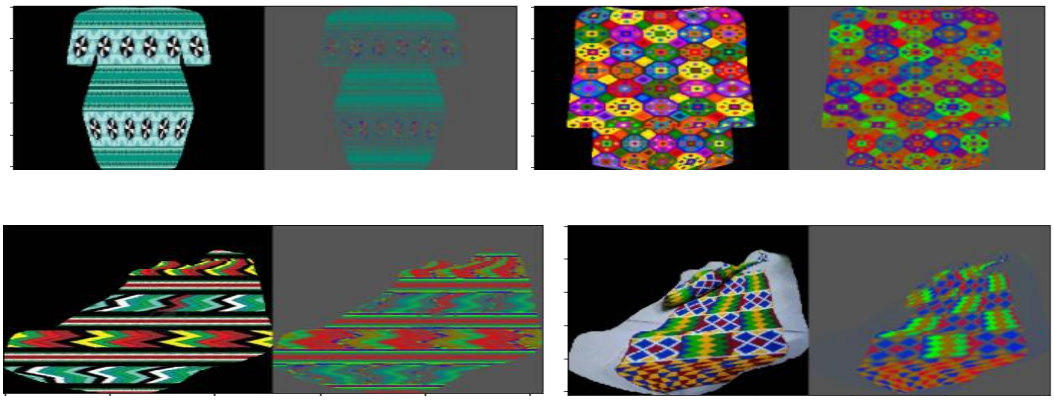


FIGURE 20) RESULTS FROM COLOR FEATURE EXTRACTION

6.2.5 Content Feature Extraction Result

Grayscale feature of an image is the best and mostly used feature to represent local structure of image and we can use them to measure similarity between images but it is not suitable when images of regions on images we want to compare vary in dimension. Texture feature is also the one used to represent local structure of images. Based on techniques used to extract texture feature it can be invariant like scale invariant. But existing texture feature extraction mechanisms are exposing things for difficulties. The content feature or local structure feature we developed looks like grayscale feature of images but slightly different. some papers tried to used grayscale which can be mean of RGB representing pixels or L-channel of Lab color space. but we identified some difficulties of using gray feature of images as content feature while trying to change color of pixels of images.

Let p1= [r=255, g=255, b=201] be pixel of image x, p2= [255,0,0] be pixel of image y. let us try to transfer color information of p2 to p1 and p3 be the new pixel generated having content information of p1 and color information of p2.

This task has been tried to be solved in different papers. The followings are some of methods used in those papers.

- ✓ L1, L2, Euclidian distance and other methods are being used
- ✓ Using feature extractors and measuring similarity between feature vectors extracted.

- ✓ Measuring similarities after transformation like mean of pixels of content and color space transformations

Mean of p1 channels is equal to 237.0 and can represent content feature of that pixel and color of p2 is pure red. When we think about the mathematics behind generating p3 with mean of its channels equals with 237.0 and pure red it is hard. Mean or average of pure red pixel is 85.0 which show that impossibility of having pure red pixel with average of its channels is greater than 85.0.

So we have understood that there may be change on average or mean of channels when color transferring or color involving style transfer task is tried.

We extracted color information of images by taking ratio of each channels of each pixels to their sum. Since average of channels of each pixels of the color feature we have extracted is nearly equal, which is 0.333, our color feature does not contain and information about local structure of the image. We have extracted color features of both style reference images and generated images. But to measure structural similarities of generated images and content reference images we extracted content feature from gray feature image of reference images transformed according to the maximum average or mean of pixels possibly generated when the corresponding pixels ratio to their sum is used. So then we calculated transformation weight from intended color feature which is ratio between gray values of pixel could be made using the corresponding color ratio calculated and gray value of pixel could be calculated from RGB pixel could be gained when possible maximum of corresponding pixel color ratio calculated.

6.3 Generated Samples by Randomly Select Saved Models

After 6 days, including electric power off times, of training 100 iterations completed. We have reached at best model selection. Let us start from images generated by randomly selected models from saved models above 30% of total iteration.

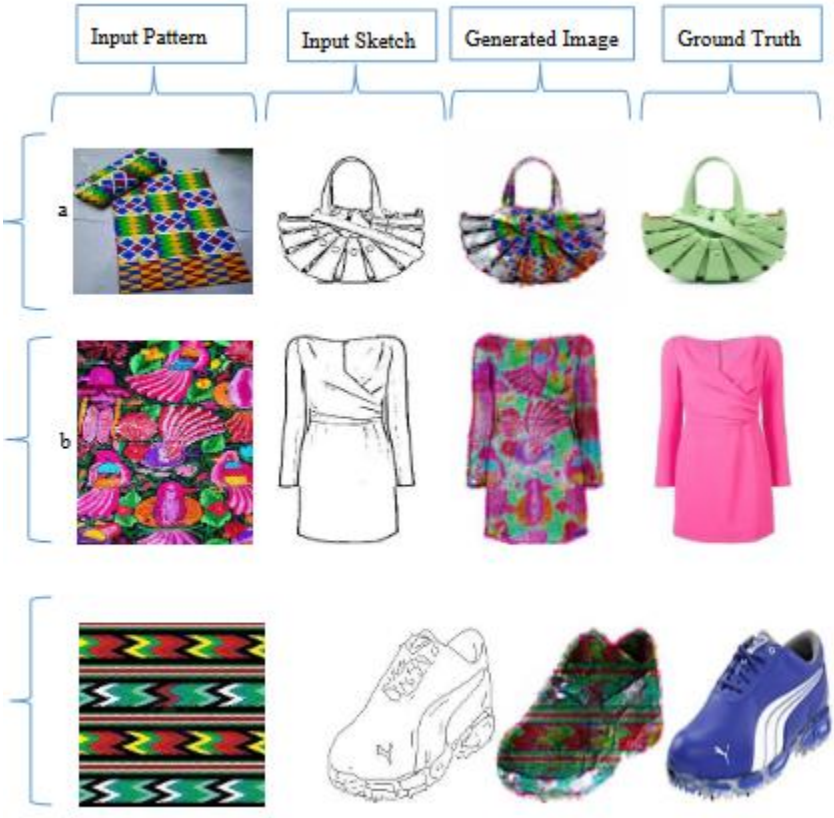


FIGURE 21) GENERATED IMAGES BY RANDOMLY SELECTED MODELS

As shown in Figure 21) generated images by randomly selected models shows generated images by randomly selected models differ in quality which is resulted from state of the selected model. For example image generated for shoes (Figure 20 c) is low in quality while quality of generated bag and dress (a and b) is good. Because of this we have to measure or test our saved models and select the best model. For this reason, we have measured performance of our saved models in different angles. Inception Score, Human evaluation, whether or not input features are used to control their corresponding intended target, other criteria's like application of our models in local cultural patterns and loss functions are used to help us to reach at the decision.

6.4 Discussion on Results of Controlling Features

To check whether or not the intended target or objective is met, we have used different methods and analyzed and discussed our results. Under succeeding subtitles how input features are translated to give us generated images, application of our model in local and global problems of culture reflecting fashion design and other results are discussed. Also, progress of our model on generation of images during training is putted on Appendix 11: Progression on Bags Image Generation Progress on Bags' image generation, Appendix 10: Progress on Dresses Image Generation Progress on Dresses' images generation and Appendix 9: Progression on Shoes Image Generation Progress on Shoes' image generation can be referred.

6.5 Model Pattern Interpretation

To see whether or not input pattern is interpreted correctly, we applied the same pattern image to different sketch. The intention was to check if generated images using sketch of different objects but the same input pattern share similar color information or color pattern. We selected one item from each class, bags dresses and shoes, and then applied the same pattern. We did the process multiple times and color information of the three generated images using the same pattern are similar which shows that our model clearly understood the purpose of input pattern and interpreted correctly. ***Figure 22) Pattern Input Interpretation by*** Model shows results from three steps and for more results check Appendix 13: Input Pattern Controlling.

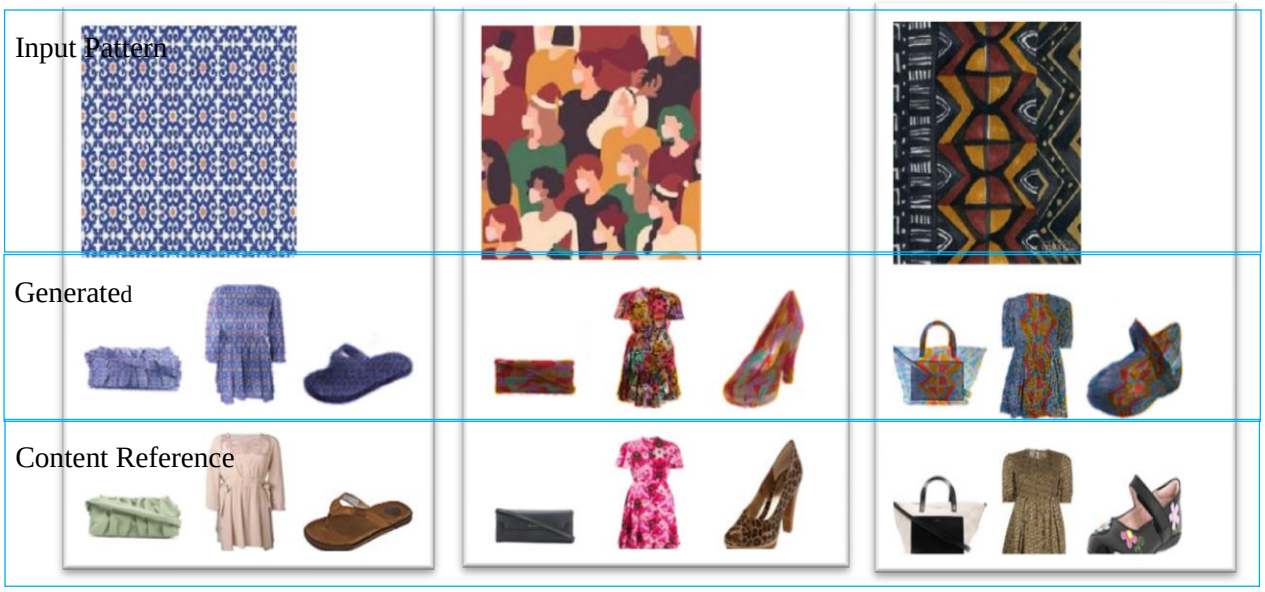


FIGURE 22) PATTERN INPUT INTERPRETATION BY MODEL

6.6 Model Input Sketch Interpretation

The target of input sketch was to control local structure generated image. To see whether or not our generated image local structure translated from sketch or sketch inputted controlled the structural properties of the generated image we tested our model by having the following points in mind.

- ✓ Did existence or absence of particular region, part, or element on input sketch controlled existence or absence of its intended corresponding region, part or element on generated image?
- ✓ Is there information of content reference not to be controlled by sketch?

So then we selected pair of sketch images one with presence of a particular feature and the other with absence of the same feature and tested our model**.** Figure 23) Input Sketch Interpretations by our Model shows that our model effectively understood as the sketch is to control local structure of generated image. We added two different shapes like triangle differ in their thickness to sketch. Their effect differs like thickness of lines. To see if the difference made on local structure of the generated image around the added lines or shapes, we redraw those shapes using thicker lines but equal and we got their effect around those lines is the same. For more results, please refer images under Appendix 14: Input Sketch Controlling.

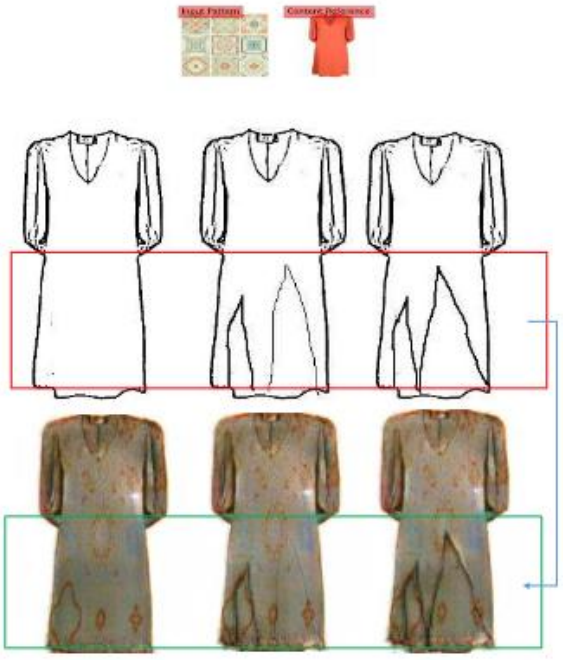


FIGURE 23) INPUT SKETCH INTERPRETATIONS BY OUR MODEL

6.7 Class Label Interpretation

To see the effect of input label we analyzed generated images, we inputted a given image in a particular class three times by keeping sketch and pattern the same but changing class labels. That means generating image with shoes sketch and bags label, shoes sketch and dresses label and shoes sketch with shoes label. For bags and dresses we repeated the same process. They are very different. Only image with correct label and sketch generated correctly. For example, during giving bag sketch as dress and shoe, generator model added unwanted part to generated image.

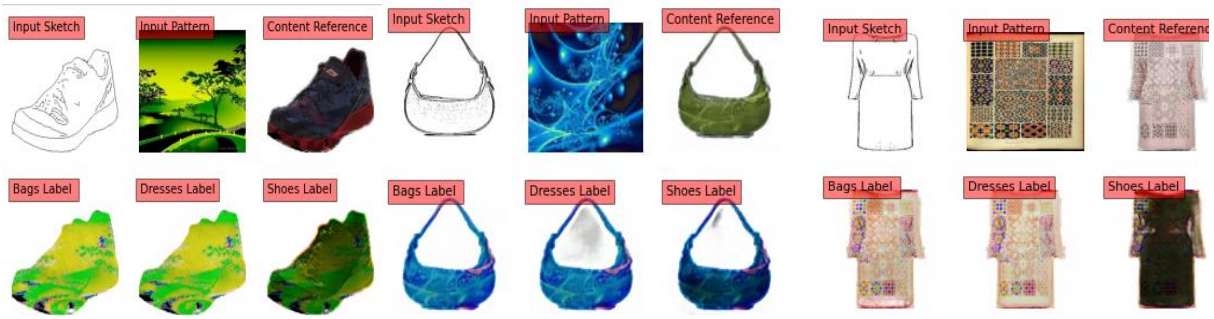
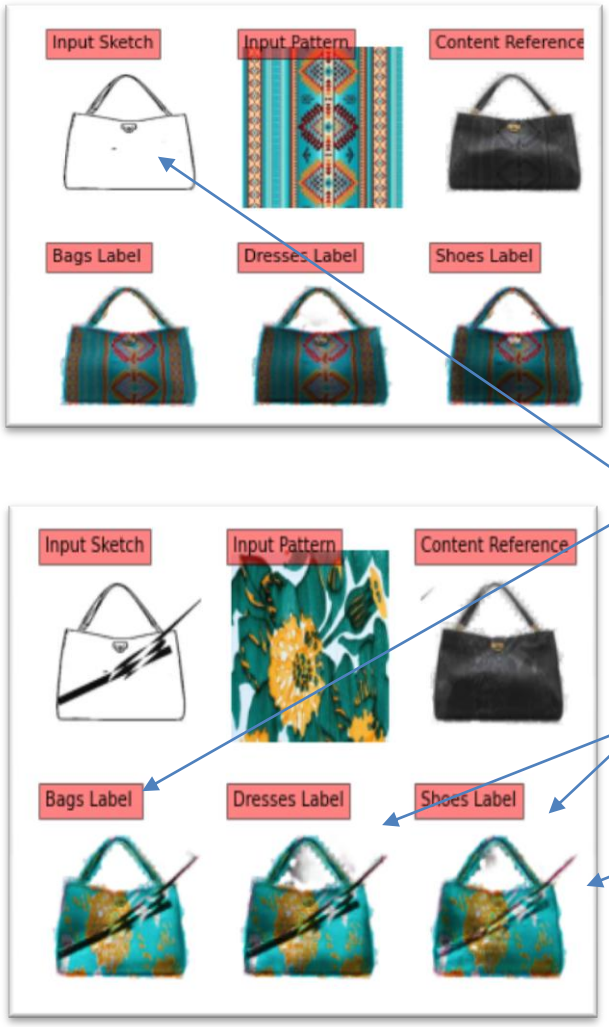


FIGURE 24) INTERPRETATION OF CLASS LABEL BY OUR MODEL

It is difficult to say our model is correct using only Figure 18 about label interpretation. The label was originally intended to control global structure or class label common attributes of items belonged to that class. The meaning of lines, shapes, dots and other information could be different across classes because shoes, bags and dresses have different parts. to answer the question that says, “did the model understood the input label as a condition to control global or common attributes of dresses or shoes or bags?”, we added some information like lines to sketch image and fed the generator model with different labels. For example, adding line to bag sketch and generate three images, the first one from bag sketch modified with dress label, the bag sketch modified with shoes label and the bag sketch modified with bag sketch. Then we analyzed how the added line is interpreted. We did this because of that our model trained with sketch and its corresponding class label which is very challenging even impossible to how lines, shapes or dots on sketch interpreted by the model.



- ✓ The input sketch did not contain many lines, dots and other information and because of the all three images generated with bag sketch and bags' label, bag's sketch and shoes' label and bag's sketch and dresses' label are not very far from each other.
- ✓ We drew lines on the original sketch
- ✓ Then we repeated the same process of generating three image using the same sketch and pattern but different labels.
- ✓ During class label of bags and dresses are used, regions around the added lines sinks
- ✓ During shoes label is used regions around the added line rises.
- ✓ This example is when using Bag

FIGURE 25) GLOBAL ATTRIBUTE CONTROLLING

The other thing we observed is that thickness of lines has different meaning in different classes. For example, during we tested our model of shoes image generation, lines of different thickness used to determine structure of surrounding regions while in dresses and bags if the width of the line is wide in addition to controlling rising and sinking of sounding region it remains by itself as region without color, very thin lines disappears and middle range wide lines control local structure of surrounding regions.

Generally using our model someone get new design simply by changing some sketch like adding lines, dots and other shaper or removing some them. This trick enables us getting completely different, interesting and many fashions from existing sketch of a particular fashion. Appendix

15: Global Feature Controlling shows more results we have got as a result of label feature controlling and feature addition and removal on input sketch image.

6.8 Results on Ethiopian Cultural Patterns

To test training using multicolored and complex cultural patterns could enable our model to be generic culture reflecting fashion designer model, we tested with local cultural pattern we have collected from local traditional cloth shops and designers. The following figure shows some of them and for more results refer Appendix 12: Performance On Ethiopian Culture Reflection.

FIGURE 26) ETHIOPIAN CULTURE REFLECTING FASHIONS

6.9 Handling Generation of Multicolored and Complex Texture Image Generation

Handling image generation with complex texture and multicolored was the problem of many fashion generation models including related works we have reviewed. We have controlled our model in the way that it can generate such type images and our model shows incredible performance.

Multiple color completely different from that of original images are generated in addition to structural complexity of their content references. The algorithm we developed and used simplified the process of handling them without having side effect on each other meaning, patterns complexity did not affect the accuracy of local structure and vice versa.

FIGURE 27) HANDLING COMPLEX TEXTURE AND MULTIPLE COLOR

6.10 Evaluation and Comparison with Baseline

6.10.1 Human Evaluation

We evaluated our models using Human Perceptual Study. First 10 voluntary peoples are selected. 10 images from the same sketch and color patterns are generated using candidate saved models. Then peoples are asked to rate generated images [0,10] according to structural similarity with content reference and color similarity with color pattern references.

TABLE 15) TABLE OF HUMAN EVALUATION

No			FashionGAN	Ours
1.	Local Structure Accuracy	Bags	6.09	8.12
		Dresses	7.07	7.90
		Shoes	8.10	8.3
2.	Color Pattern Accuracy	Bags	5.20	9.80
		Dresses	6.04	9.10
		Shoes	4.90	8.80
3.	Average		6.233333	8.67

6.10.2 Inception Score

For measuring quality of generated images and their diversity, which means ability of our generator model to generate different images from single instance of sketch we have generated 6 images from single sketch image and random pattern image Figure 28) Different Images Generated from Single Instance. Then we used our discriminator model and pre-trained InceptionV3 model to classify our generated images to their corresponding classes and their result is used to calculate inception score. To make it simple and clear we have used results from InceptionV3 pre-trained model for this report Table 16) Table of Inception Score

FIGURE 28) DIFFERENT IMAGES GENERATED FROM SINGLE INSTANCE

TABLE 16) TABLE OF INCEPTION SCORE

No	FasshionGAN	Ours
1	2.643	2.97

6.11 More Applications

The custom assisting algorithm we have developed can be used as preprocessing layer like Normalization layers and Activation layers and can be used in deep learning models to facilitate the transformation of patterns or other image to corresponding intended object area. Not only this, using the advantage of UNET architecture, means ability of down sampling and down sampling to handle relationship between pixels in images, our custom assisting algorithm can be applied during loss calculation to enforce the model to produce image transformed in such way.

7 CONCLUSION AND FUTURE WORK

7.1 Conclusion

Visualizing visual culture, which is tangible custom and belief of a particular group of people at particular time, on fashion allows people to reflect themselves. Most of visual cultures are the work of art and abstract which can be made using color mixture and complex patterns. even though GANs are rapidly changing the world of fashion the problem of generating fashion images styled with multiple colors and complex structure which in turn problem for reflecting culture on fashions. So by this paper we presented the new deep learning frame work that can design fashions in the way that culture can be reflected through them and new custom algorithm that assisted the deep learning model. We have evaluated our model and our algorithms with other models and algorithms having similar purpose with our work and incredible improvements gained are presented.

The new framework we developed can be implemented for many image to image translation and image synthesis with complex texture made color information. In addition to taking sketch as input our model can take grayscale images and colorful patterns or fabric patterns and can learn mapping or translation of both features to single image containing features from both inputs according to user need.

7.2 Future Work

U-NET architecture enables Generator models to learn relationships among pixels of an image at any coordinates. But to control the direction of generation existence of exact matching ground truth very difficult and limiting us from generating items with properties we are having in mind. Specially, during more than one reference images are used measuring similarity at instance level is impossible. So many papers have been using invariant feature extractor and exposed to many problems because of their drawbacks. So in future we promise to develop deep learning model having our algorithm as preprocessing layers like Normalization and Activation Layers that transforms raw input pattern internally. Also we will try U-NET architecture to generate images and use our algorithm as feature transformation on ground truth and then calculate loss between transformed and generated image.

8 REFERENCES

Aiyu Cui, D. M. (2021, Aug 31). Dressing in Order: Recurrent Person Image Generation for Pose Transfer, Virtual Try-on and Outfit Editing. Retrieved from arXiv:2104.07021v2

Andrew Zhai, D. K. (2017). Visual Discovery at Pinterest. *International World Wide Web Conference Committee.* Perth, Australia: CV. Retrieved from arXiv:1702.04680v2 [cs.CV] 25 Mar 2017

Anita Rácz, D. B. (2021). Effect of Dataset Size and Train/Test Split Ratios in QSAR/QSPR Multiclass Classification. *Molecules*, 1111. doi:https://doi.org/10.3390/molecules26041111

Debi Robersona, J. D. (2005). Evidence for the cultural relativity hypothesis. *Cognitive Psycology*. doi:10.1016/j.cogpsych.2004.10.001

Delwin T. Lindseya, a. A. (2008). World Color Survey color naming reveals universal motifs and their within-language diversity. *PNAS*.

Diederik P. Kingma, a. J. (2017). ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. *ICLR 2015*. Retrieved from arXiv:1412.6980v9 [cs.LG] 30 Jan 2017

Domicele Jonauskaite, A. A.-A. (2020). Universal Patterns in Color-Emotion Associations Are Further Shaped by Linguistic and Geographic Proximity. *ASSOCIATION FOR PSYCHOLOGICAL SCIENCE*, 1245–1260. doi:doi.org/10.1177/09567976209488

Efros, P. I.-Y. (2017). Image-to-Image Translation with Conditional Adversarial Networks. *IEEE conference on computer vision and pattern recognition*, (pp. 1125–1134).

Gökhan Yildirim, C. S. (2018, Jun 20). Disentangling Multiple Conditional Inputs in GANs. *Computer Vision and Pattern Recognition*, 5. Retrieved from https://arxiv.org/abs/1806.07819

Gökhan Yildirim, C. S. (2018, Jun 20). Disentangling Multiple Conditional Inputs in GANs. *arXiv*. doi:arXiv:1806.07819v1 [cs.CV]

Haijun Zhang, Y. S. (2018). ClothingOut: a category-supervised GAN model for clothing segmentation and retrieval. *The Natural Computing Applications Forum*. doi:https://doi.org/10.1007/s00521-018-3691-y

Hughes, K. S. (2013). Possession Rituals of the Digital Consumer: a Study of Pinterest. *E - European Advances in Consumer Research.10*, pp. 47-50. Association for Consumer Research. Retrieved from http://www.acrwebsite.org/volumes/1013718/volumes/v10e/E-10

INTRODUCTION TO SOCIOLOGY: UNDERSTANDING AND CHANGING THE SOCIAL WORLD. (n.d.). PRESSBOOKS.

J. Huang, R. F. (2015). Cross-domain image retrieval with a dual attribute-aware ranking network. *IEEE International Conference on Computer Vision*, (pp. 1062–1070).

Jiaming Xu, X. G. (2019). POG: Personalized Outfit Generation for Fashion Recommendation. *Proceedings ofŠthe 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (p. 9). Anchorage, AK, USA: KDD.

Kristen Vaccaro, S. S. (2016). The Elements of Fashion Style. *ACM*.

Linlin Liu a, H. Z. (12 March 2019). Toward AI fashion design: An Attribute-GAN model for clothing match. *Neurocomputing*.

Liu, S. Y. (2016). Fashion Landmark Detection in the Wild. *ECCV* .

Olaf Ronneberger, P. F. (2015, May 8). U-Net: Convolutional Networks for Biomedical Image Segmentation. *CV*. Retrieved from arXiv:1505.04597v1 [cs.CV] 18 May 2015

Phillip Isola, J.-Y. Z. (2017, Nov 26). Image-to-Image Translation with Conditional Adversarial Networks. *CVPR*. doi:arXiv:1611.07004v3

Plate, S. B. (2015, July). Visual Arts and Culture. *Research Gate*. doi:10.13140/RG.2.1.2843.5041

Pozzo, B. (2020, February 12). Fashion between Inspiration and Appropriation. *MDPI*.

Qiang Wu, B. Z. (2021). ClothGAN: generation of fashionable Dunhuang. *Connection Science*. doi:10.1080/09540091.2020.1822780

Ramirez, M. (2019, July 30). Cultural patterns in product design ideas:comparisons between Australian and Iranian student concepts. *ResearchGate*.

Riello, G. L. (2008). EAST & WEST: TEXTILES AND FASHION IN EARLY MODERN EUROPE. Retrieved from http://chnm.gmu.edu/jsh/index.html

Ruggero Pintus, K. P. (2016). A Survey of Geometric Analysis in Cultural Heritage. *COMPUTER GRAPHICS forum*. doi:10.1111/cgf.12668

Seyed Omid Mohammadi, A. K. (March 202). Smart Fashion: A Review of AI Applications in the Fashion & Apparel Industry. *arx*, 99.

Shizhan Zhu, S. F. (n.d.). Fashion Synthesis with Structural Coherence. *ICCV*.

Thomas Ziegler, J. B. (2020, Mar 26). Fashion Landmark Detection and Category Classification for Robotics. *arXiv*. Retrieved from arXiv:2003.11827v1

Wang-Cheng Kang, C. F. (2017). Visually-aware fashion recommendation and design with generative image models. *IEEE International Conference on Data Mining* (pp. 207–216). IEEE.

Xintong Han, X. H. (2019). Clothflow: A flow-based model for clothed person generation. *International Conference on Computer Vision* (pp. 10471–10480). IEEE/CVF.

Y. Ge, R. Z. (2019). A Versatile Benchmark for Detection, Pose Estimation, Segmentation and Re-Identification of Clothing Images. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* Retrieved from http://arxiv.org/abs/1901.07973

Z. Liu, P. L. (2016). Deepfashion: powering robust clothes recognition and retrieval with rich annotations. *Conference on Computer Vision and Pattern Recognition* (pp. 1096–1104). IEEE.

9 APPENDICES

Appendix 1: Data Generator Class Implementation

```
class DataGenerator(tf.keras.utils.Sequence):
    def __init__(self, fashions, patterns, batch_size=2, shuffle=True):
        super().__init__()
        self.fashions = fashions
        self.patterns = patterns
        self.batch_size = batch_size
        self.shuffle = shuffle
        self.key_array = np.arange(self.fashions.shape[0], dtype=np.uint32)
        self.class_indices={'ba':0,'dr':1,'sh':2}
        self.on_epoch_end()

    def __len__(self):
        return int(self.batch_size)

    def __getitem__(self, index):
        keys = self.key_array[index*self.batch_size:(index+1)*self.batch_size]

        fash = self.fashions[keys]
        patt = np.random.choice(self.patterns,self.batch_size)

        pattern_img=np.empty([self.batch_size,256,256,3],dtype=np.float32)
        pattern_img_=np.empty([self.batch_size,256,256,3],dtype=np.float32)
        y=np.empty([2,self.batch_size,256,256,1],dtype=np.float32)

        labels=self._extract_labels(fash)
        for i in range(self.batch_size):
            img_0=self._read_image(fash[i])
            y[0,i]=img_0[0]
            img_1=self._read_image(patt[i],1)
            color_img,th=self.extract_color(self._mask_pattern(img_0[2],img_1))
            pattern_img[i]=color_img
            y[1,i]=img_0[1]*th
        return labels,pattern_img,y/255
```

Appendix 2: Custom Assisting Algorithm(WAPM) Implementation

```
def the_assisting_algorithm(self,mask_img,pattern_img):
        sh=np.squeeze(maskt_img,axis=-1)
        sh=np.where(sh>10,1,0)
        pl=np.ones_like(pattern_img)
        w=np.count_nonzero(sh,axis=0)
        w=w[w>0]
        h=np.count_nonzero(sh,axis=1)
        h=h[h>0]
        new_pat=tf.image.resize(np.expand_dims(pattern_img,axis=0),(h.shape[0],w.shape[0]))
        rows=tf.split(new_pat,h.shape[0],axis=1)
        vals=[]
        for k in range(len(rows)):
            vals.append(tf.squeeze(tf.squeeze(tf.image.resize(rows[k],(1,h[k])),axis=0),axis=0).numpy())
        pl[np.where(sh)]=K.concatenate(vals,axis=0).numpy()
        return pl
```

Appendix 3: Implementation for Color Feature Extraction

```
def extract_color(self,img):
    img=tf.cast(img,tf.float32)
    sm=tf.reduce_sum(img,axis=-1,keepdims=True)
    color_img=tf.math.divide_no_nan(img,sm)
    max_max=tf.math.divide_no_nan(1.0,tf.reduce_max(color_img,axis=-1,keepdims=True))
    max_ratio=tf.reduce_mean(color_img*max_max,axis=-1,keepdims=True)
    return color_img.__array__(),max_ratio.__array__()
```

Appendix 4: Implementation of Down Sampling Block:

```
def downsample(channels, kernels, strides=2,norm=True, activation=True, dropout=False):
    initializer = tf.random_normal_initializer(0.0, 0.02)
    block = tf.keras.Sequential()
    block.add(layers.Conv2D(channels, kernels,strides=strides,
                            padding='same',use_bias=False,kernel_initializer=initializer))
    if norm:
        block.add(layers.BatchNormalization())
    if activation:
        block.add(layers.LeakyReLU(0.2))
    if dropout:
        block.add(layers.Dropout(0.5))
    return block

```

Appendix 5: Implementation of Up Sampling Block:

```
def upsample( channels, kernels, strides=1,norm=True, activation=True, dropout=False):
    initializer = tf.random_normal_initializer(0., 0.02)
    block = tf.keras.Sequential()
    block.add(layers.UpSampling2D((2,2)))
    block.add(layers.Conv2D(channels, kernels,strides=strides, padding='same',use_bias=False,
                            kernel_initializer=initializer))
    if norm:
        block.add(InstanceNormalization())
    if activation:
        block.add(layers.LeakyReLU(0.2))
    if dropout:
        block.add(layers.Dropout(0.5))
    return block
```

Appendix 6: UNET Implementation

```
def unet():
    DIM=64
    inp=tf.keras.layers.Input(shape=(256,256,4),name="concatenate_9")
    down1 = downsample(DIM, 4, norm=False)(inp) # 128
    down2 = downsample(2*DIM, 4)(down1) # 64
    down3 = downsample(4*DIM, 4)(down2) # 32
    down4 = downsample(4*DIM, 4)(down3) # 16
    down5 = downsample(4*DIM, 4)(down4) # 8
    down6 = downsample(4*DIM, 4)(down5) # 4
    down7 = downsample(4*DIM, 4)(down6) # 2

    up6 = upsample(4*DIM, 4, dropout=True)(down7) # 4,4*DIM
    concat6 = layers.Concatenate()([up6, down6])
    up5 = upsample(4*DIM, 4, dropout=True)(concat6)
    concat5 = layers.Concatenate()([up5, down5])
    up4 = upsample(4*DIM, 4, dropout=True)(concat5)
    concat4 = layers.Concatenate()([up4, down4])
    up3 = upsample(4*DIM, 4)(concat4)
    concat3 = layers.Concatenate()([up3, down3])
    up2 = upsample(2*DIM, 4)(concat3)
    concat2 = layers.Concatenate()([up2, down2])
    up1 = upsample(DIM, 4)(concat2)
    concat1 = layers.Concatenate()([up1, down1])
    return Model(inputs=inp,outputs=concat1,name='UNET')
```

Appendix 7: Generator Model Implementation

```
def build_generator(image_shape=(256,256,3),num_classes=3):
    DIM = 64
    input_image = layers.Input(shape=(256,256,1),name='Sketch_Input')
    input_pattern = layers.Input(shape=image_shape,name='Pattern_Input')

    input_label = layers.Input(shape=(1,),name='label_input')

    embedding = layers.Embedding(num_classes,256*256*1)(input_label)
    embedding = layers.Reshape((256,256,1))(embedding)

    x = layers.Multiply()([input_image, embedding])
    x = layers.Concatenate()([x,input_pattern])
    UNET=unet_(x)
    output_image = sigmoid(upsample(3, 4, norm=False,activation=None)(UNET))
    return Model(inputs=[input_label,input_image,input_pattern],outputs=output_image,name='Generator_Model')
```

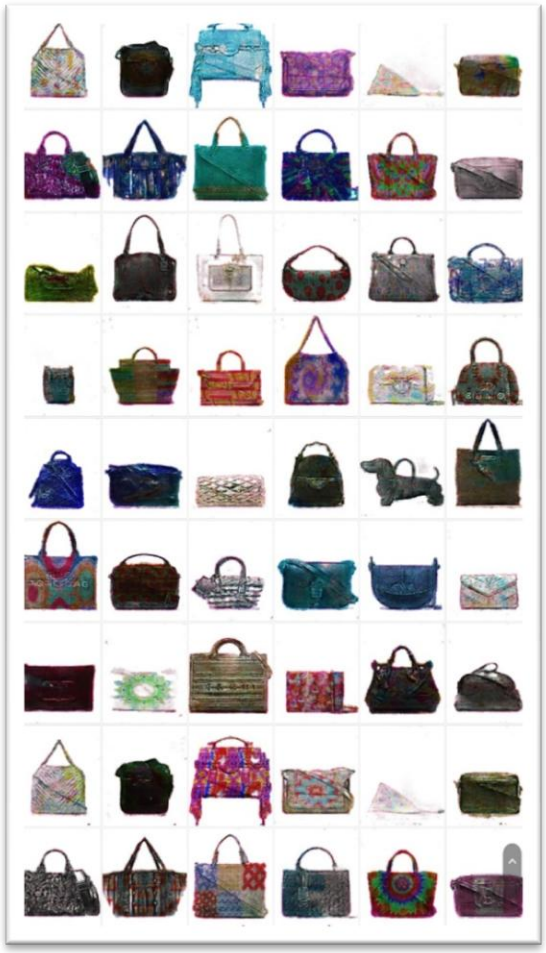
Appendix 8: Discriminator Model Implementation:

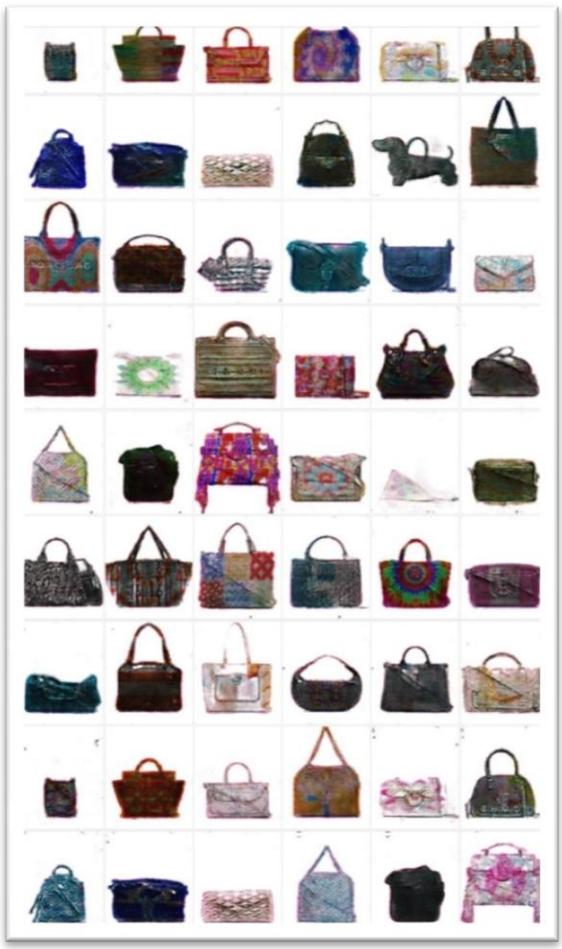
```
def build_discriminator(image_shape=(256,256,1)):
    DIM = 64
    input_image_A = layers.Input(shape=(256,256,1),name='sketch_input')
    input_image_B = layers.Input(shape=image_shape,name='gray_input')
    x = layers.Concatenate()([input_image_A,input_image_B])
    x = downsample(DIM, 4, norm=False)(x)
    x = downsample(2*DIM, 4)(x)
    x = downsample(4*DIM, 4)(x)
    x = downsample(8*DIM, 4, strides=1)(x)
    x=layers.Flatten()(x)
    label_output=layers.Dense(3,activation='softmax', name='Class_Label_Output')(x)
    critic_output = layers.Dense(1, activation='sigmoid',name='Real_Fake_Output')(x)
    return tf.keras.models.Model([input_image_A, input_image_B],
                                 outputs=[critic_output,label_output],name='Discriminator_Model')

```

Appendix 9: Progression on Shoes Image Generation

Appendix 10: Progress on Dresses Image Generation

Appendix 11: Progression on Bags Image Generation

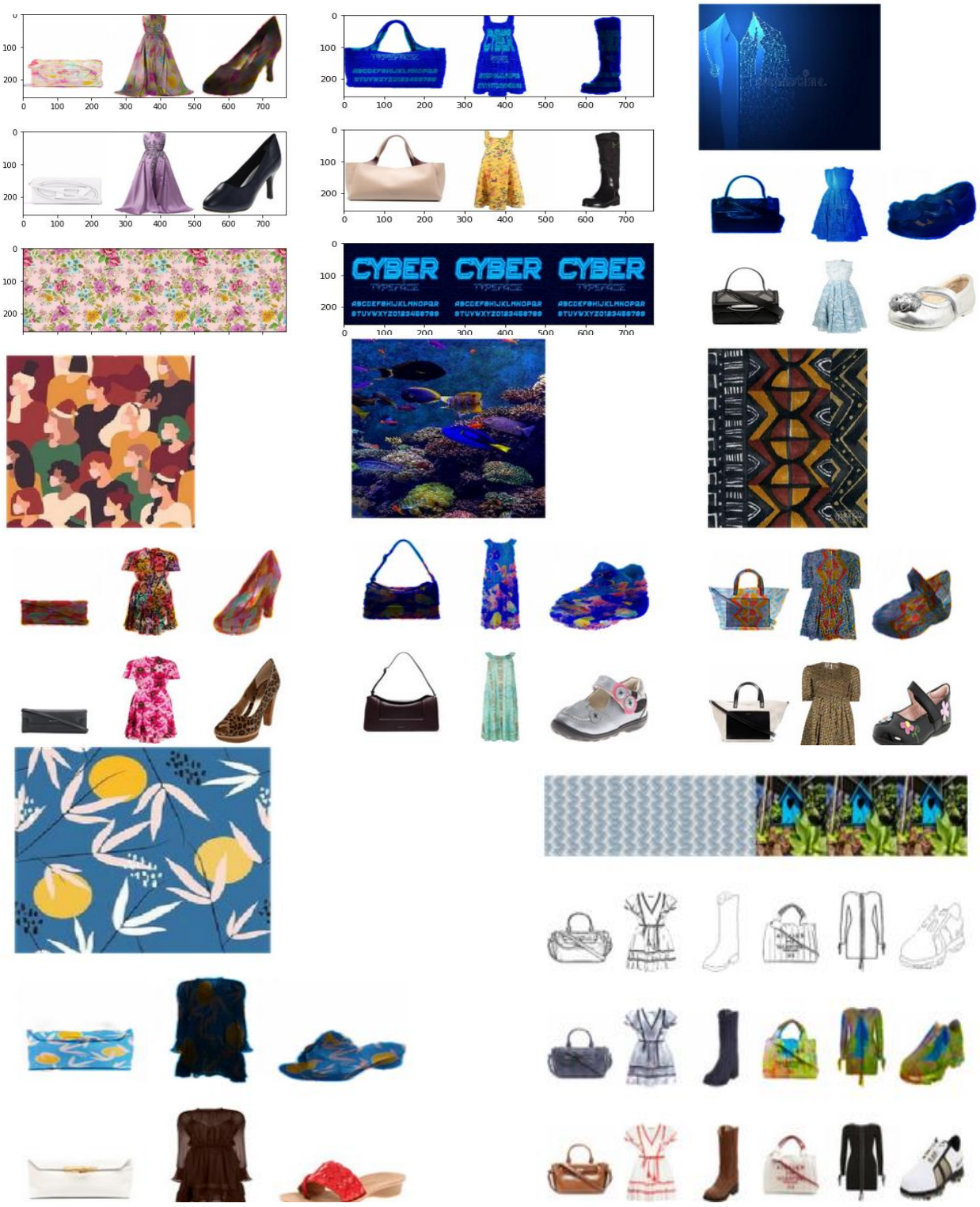




Appendix 12: Performance On Ethiopian Culture Reflection

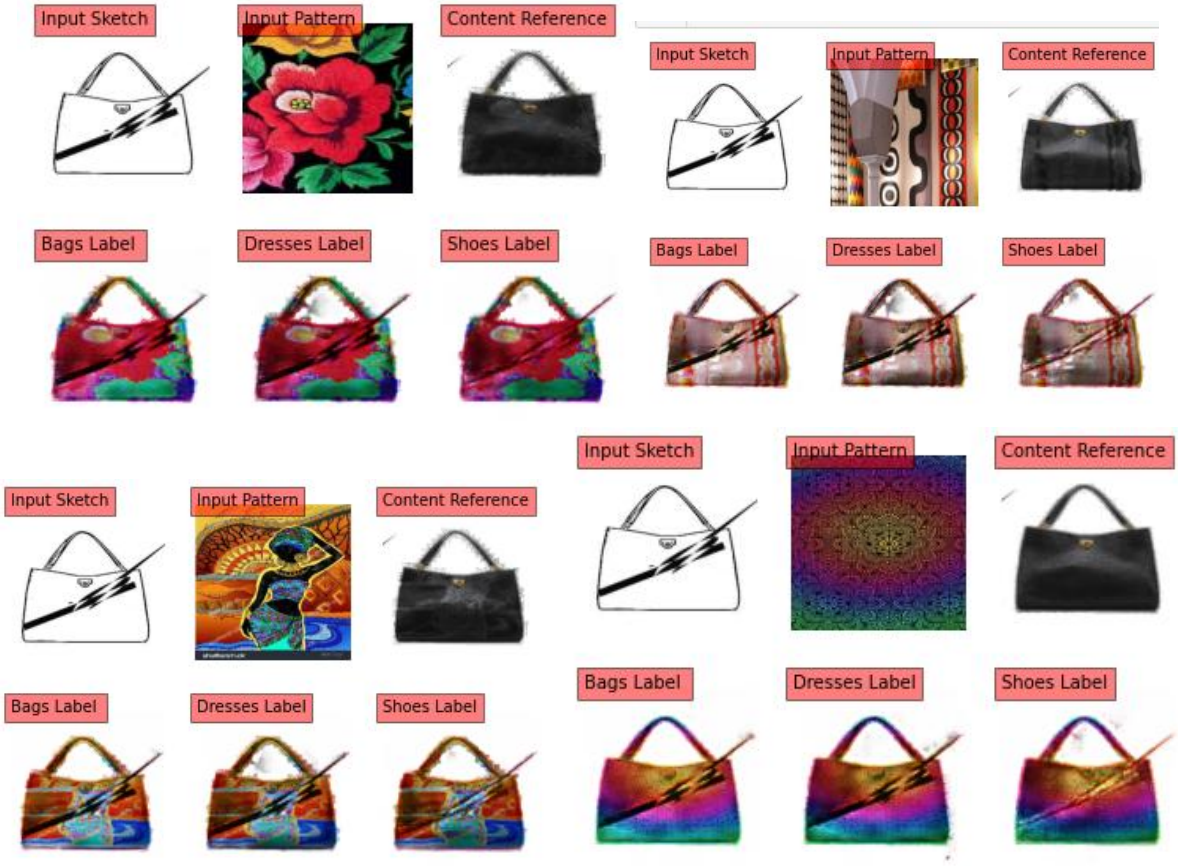

shutterstock.com · 1970734847

Appendix 13: Input Pattern Controlling

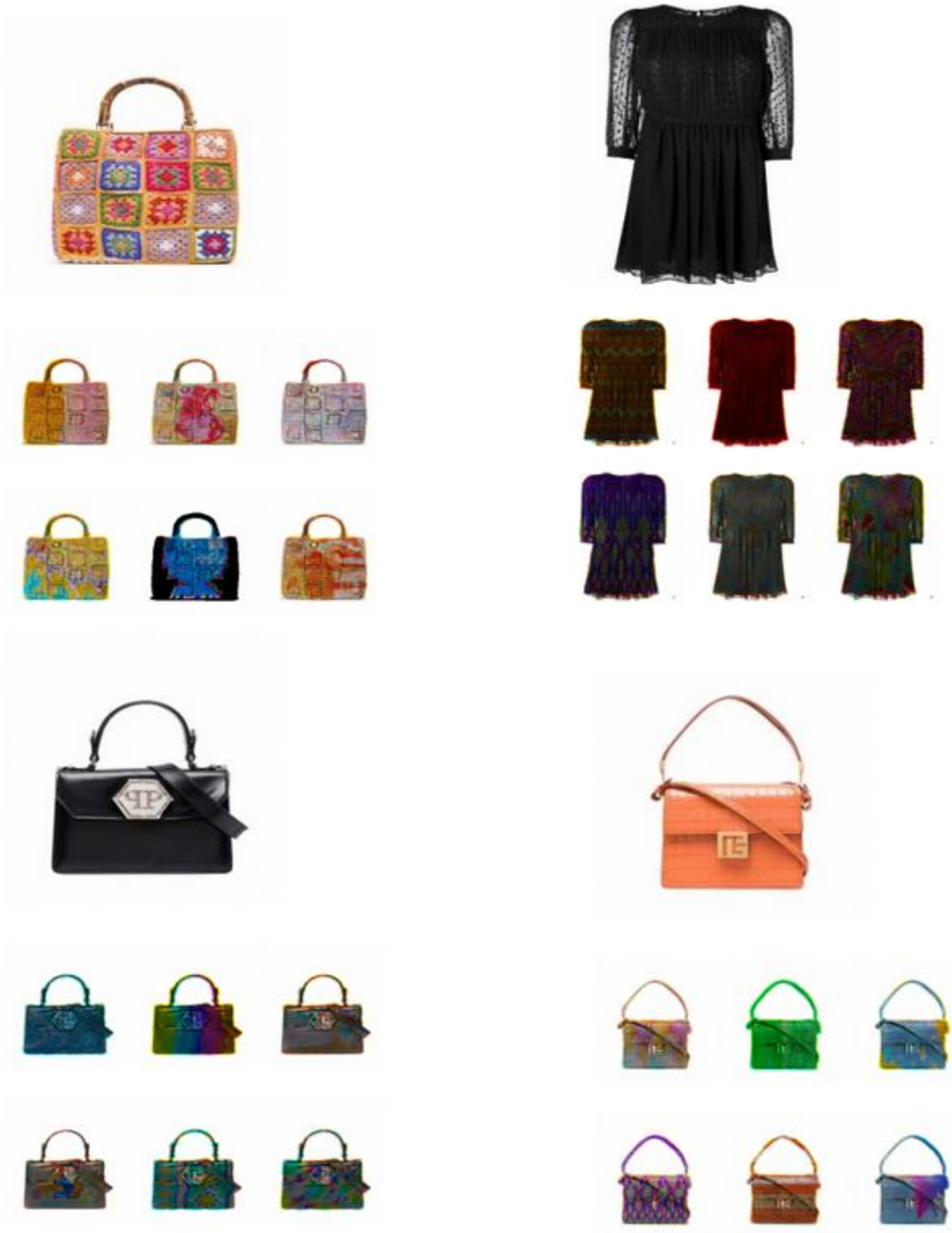


Appendix 14: Input Sketch Controlling

Appendix 15: Global Feature Controlling


Input Sketch
Input Pattern
Content Reference
Input Sketch
Input Pattern
Content Reference
Bags Label
Dresses Label
Shoes Label
Bags Label
Dresses Label
Shoes Label
Input Sketch
Input Pattern
Content Reference
Input Sketch
Input Pattern
Content Reference
Bags Label
Dresses Label
Shoes Label
Bags Label
Dresses Label
Shoes Label

Appendix 16: Diverse Image Generation Examples




SALE

Appendix 17: Generator Loss Visualization