Realistic Face Generation using a Textual Description

A MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF DEGREE
OF

BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY

Submitted by:

Anukriti (Roll No: 2K17/IT/027)
Anurag Mudgil (Roll No: 2K17/IT/029)
Nakul Dodeja (Roll No: 2K17/IT/074)

Under the supervision of

Dr. DINESH KUMAR VISHWAKARMA

Associate Professor

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

DECEMBER, 2020

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

We, Anukriti (2K17/IT/027), Anurag Mudgil (2K17/IT/029) and Nakul Dodeja (2K17/IT/074), students of B.Tech. (Information Technology), hereby declare that the project Dissertation titled "Realistic Face Generation using a Textual Description" which is submitted by us to the Department of Technology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Date: 7th December, 2020 Anukriti (2K17/IT/027)

Place: New Delhi Anurag Mudgil (2K17/IT/029)

Nakul Dodeja (2K17/IT/074)

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "Realistic Face Generation using a Textual Description" which is submitted by Anukriti (2K17/IT/027), Anurag Mudgil (2K17/IT/029) and Nakul Dodeja (2K17/IT/074) [Information Technology], Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Date: 7th December, 2020

Dr. Dinesh Kumar Vishwakarma

Place: New Delhi

SUPERVISOR

(Associate Professor,

Department of Information Technology)

ABSTRACT

A lot of research is going on in the field of GANs and their ability to generate photorealistic images from their textual descriptions. Still the images generated by existing tasks focus primarily on either generating birds or flowers from their descriptions. In this report, we have successfully implemented a sketch-refinement technique for human face creation from their text descriptions. This problem has a wide variety of industrial applications as well such as creating comics automatically, assisting a movie creator to generate frames, art creation assisting an artist to generate sketches and even for educational purposes. Thus, keeping the need for such a system in mind, we propose a two-staged StackGAN architecture obtained from deep convolutional neural networks. We convert features of faces like blonde hair, arched eyebrows into an image of actual person with these features. Not limiting to this we will also convert facial expressions like wide smile, happy face from its textual form to its corresponding image of the person with same expressions. The generation of a high resolution 256x256 image using captions provided in CelebA dataset makes it a valuable contribution in the field of research. Further, we successfully obtained an inception score of 4.04 ± 0.05 over 10 iterations of evaluation and have shown promising results.

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I am very thankful to Dr. Dinesh Kumar Vishwakarma (Associate Professor, IT) and all the faculty members of the Department of Information Technology of DTU. They all provided us with immense support and guidance for the project.

I would also like to express my gratitude to the University for providing us with the laboratories, infrastructure, testing facilities and environment which allowed us to work without any obstructions.

I would also like to appreciate the support provided to us by our lab assistants, seniors and our peer group who aided us with all the knowledge they had regarding various topics.

Date: 7th December, 2020

Place: New Delhi

Anukriti (2K17/IT/027)

Anurag Mudgil (2K17/IT/029)

Nakul Dodeja (2K17/IT/074)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Explanation
BERT	Bidirectional Encoder Representation from Transformers
CGAN	Conditional Generative Adversarial Network
CNN	Convolutional Neural Network
DCGAN	Deep Convolutional Generative Adversarial Network
DC-IGN	Deep Convolution Inverse Graphic
DCNN	Deep Convolution Neural Network
DR-GAN	Disentangled Representation Learning Generative Adversarial Network
GAN	Generative Adversarial Network
GB	Giga Bytes
HDGAN	High Definition Generative Adversarial Network
IS	Inception Score
KL divergence	Kullback-Leibler divergence
LFW	Labelled Faces in the Wild
PIM	Pose Invariant Model
RNN	Recurrent Neural Network
SBERT	Sentence Bidirectional Encoder Representation from Transformers
SD	Standard Deviation
SGVB	Stochastic Gradient Variational Bayes
TP-GAN	Two-Pathway Generative Adversarial Network
VAE	Variational Auto Encoder
VCG	Vickrey-Clarke-Groves

CHAPTER 1

INTRODUCTION

The following chapter provides an introduction to the conducted research. Section 1.1 provides an overview of the problem domain and Section 1.2 presents our problem statement.

1.1 OVERVIEW

The ability to imagine things based on their physical description reflects the intellectual capabilities of human beings. Whenever we read a novel we are always curious about how the characters look in reality and we often imagine the appearances of the characters; imagining the complete person is still viable but getting description of some important details is still a daunting task. When a book is converted into a movie it is very important that the cast matches the actual description of characters making it more realistic for the audience.

Apart from it sketch artists often have to draw sketches of people that they have not seen in their life just based on the textual description provided to them. This finds immense use in tracking criminals and finding lost people. Thus generating photo realistic images of people based on their textual description is a problem of critical significance. It has a variety of applications from casting in media to designing faces of actual people to find them. There is also immense potential for usage in the fields of computer aided design and editing of photographs.

In this work we focus on image generation that means making realistic images based on their textual description which aims to form the faces of people based on their features provided as description. These features may include color of eye, shape of nose, hair style, complexion etc. Although some work is done to get images of flowers and birds based on their description only a few have addressed the topic of generating faces of actual humans based on their description.

1.2 PROBLEM STATEMENT

Image captioning is one of important problem of machine learning referred to as description of image in text according to the features of the image and properties observed. It has various uses like virtual assistance, for people with visual impairment etc.

Vast amount of research has been done in the field of image captioning and it is finding its implementation in commercial products like Google photos. Text to Image generation process the inverse of the process of captioning of images that possesses its own variety of important uses like Criminal Face Reconstruction, Face Generation of characters in a story etc. and this field is still not much explored by the researchers.

Prevailing ways for conversion of textual data to images mainly focus on generating images of object, flowers or birds and only a few have addressed the issue of generating images of actual faces from the description. The ones which are based on generation of faces just reflect the basic meaning of the description and fail to show the details and vivid object parts. While image synthesis is in itself an arduous task in Computer Vision, generating images of high resolution is an even more difficult task because of possible lack of overlay between the substructures of distributions of an implied model and that of natural images in pixel space containing a large number of dimensions. With an increase in resolution further, the problem become more severe and starts giving non-sensical outputs.

The primary intention of this research is Text to Face Generation - generating 256 X 256 resolution images of faces of people on the basis of the textual description provided. We convert features of faces like blonde hair, arched eyebrows into an image of actual person with these features. Not limiting to this we will also convert facial expressions like wide smile, happy face from its textual form to its corresponding image of the person with same expressions.

CHAPTER 2

LITERATURE REVIEW

The following chapter presents a literature review corresponding to the domain of our problem statement.

Multimodal deep learning [1] involves data estimation in a mode by means of provision of data in a separate mode. Text to Image generation is an illustration of the same.
DC-GAN [2] employed an end-to-end conditional GAN architecture to produce 64x64 images using features derived from an RNN. 256x256 images synthesized by a dual-stage GAN, known as StackGAN [3], 512x512 images synthesized by an architecture comprising of nested discriminators in hierarchy, known as HDGAN [4] and attention-harnessing AttnGAN [5] are some of the more recent developments in the field of image synthesis by means of descriptive text.

Text to Face generation is a related, under-researched problem that involves synthesis of human face from the provided textual description of facial features of an individual.
Face2Text [6] provides a comprehensive dataset that amalgamates the nature of labels (facial, inferred personal and emotional) present in old datasets and renders rich, well-defined labels of differing complexity in terms of syntax and semantics . However, the small size of the dataset implies that the model may remember the entire dataset and produce pseudo-good results. This is a contributing factor towards the unaddressed nature of the problem. LFW [7] and CelebA[8] are large-scale datasets possessing images and their corresponding labels. The labels in these datasets contain information pertaining to physical attributes like, face shape and skin tone and personal attributes like, sex and age.

Prior to the advent and subsequent surge in popularity of GAN [9], [10], deep convolutional neural networks [11] and variational auto encoder [12] were employed in the field of face synthesis. An attribute-oriented solution proposed by Li et al., [13] aimed to preserve the identity of facial image by employing a combination of VCG-network and Gradient Descent algorithm. It is constrained by the possibility of facial synthesis via simple features solely. Disentangled representation is used in DC-IGN [14], alongside

SGVB algorithm (VAE) for the task of face synthesis. But computational weakness stemming from dealing of uni-attribute per batch and requirement of labeled dataset renders it infeasible.

Rapid development in GANs and their conditional variants are responsible for much of the gradual growth in the quality of generated images. TP-GAN [15] proposes a GAN architecture based on dual pathway. It provides a realistic generation of the frontal view via local and global feature details. However, there is a need of large, labeled front face dataset for the same. By employing pose code, instead of actual physical features, adjustment of the head was carried out in DR-GAN [16]. PIM [17] improves upon TP-GAN and DR-GAN by carrying out unsupervised training in the presence of a deep architecture.

A completely trainable GAN proposed by Khan et al.,[18] is capable of harnessing descriptions for image generation but its performance in terms of actual synthesis and not mere matching from given dataset remains untested. FaceID-GAN [19] involved a competition among the generator, discriminator and identity classifier in terms of preservation and quality of image. It was able to achieve facial modification via expression attributes. However, self-specification of features is restrictive. To overcome this, FaceFeat-GAN [20] involved a dual stage process that produces diverse, synthesized features and identity preserving, high quality images from these features in the two respective stages. However, a broader-scoped, identity-preserving, accurate, synthesis of face from input textual description remains largely unaddressed.

CHAPTER 3

BACKGROUND

The following chapter provides the background to the conducted research. Section 3.1 discusses discriminative algorithms. Section 3.2 discusses generative algorithms. Section 3.3 discusses generative adversarial networks and their architecture. Section 3.4 discusses conditional GAN and their architecture. Section 3.5 discusses word and sentence embeddings along with Sentence BERT architecture.

3.1 DISCRIMINATIVE ALGORITHMS

There has been lots of research on discriminative algorithms in recent times. They are the algorithms that basically focus on distinguishing or classifying data instances. More precisely, they help finding the labels of data on the basis of their features. Discriminative focuses on finding the conditional probability $p(Y \mid X)$.

For example, On the basis of a photograph deciding whether it is a cat or a dog.

3.2 GENERATIVE ALGORITHMS

The reverse of discriminative models is known as generative models. That means on the basis of the labels we have to find the feature set for the given label. They include distribution of data itself and generate the data from random noise or uniform distribution. The main focus of generative models is to calculate Joint Probability $p(X, Y)$. In case labels are absent, they aim to find $P(X)$.

For example, on the basis of input cat or dog generating a relevant picture of the animal.

3.3 GENERATIVE ADVERSARIAL NETWORKS

General Adversarial Networks (abbreviated as GAN) are most ordinarily known examples of generative models that were introduced by Ian Goodfellow and his colleagues in the year 2014. According to, Yann Le Cun (Director of AI at Facebook),

GANs are representative of the most riveting innovations in the field of machine learning in the past decade.

3.3.1 Architecture of GAN

The main idea behind GAN is two neural networks contesting with each other to the other. These two neural networks are called generator(the artist) and discriminator(the art critic). Generator is a neural-network model that aims to synthesize images that are fake in nature by means of random noise or uniform data. Then these fake images by generator and training images act as input to the Discriminator which is a traditional neural network whose main aim is to discern between the real and fake images. Thus the primary goal of the Generator is to fool the discriminator and that of discriminator is to prevent itself from getting fooled. The discriminator then provides feedback to the generator to make images more close to the real ones. When training begins the generator forms obviously fake instances and discriminators easily distinguish between them but as the training continues both generator and discriminator becomes stronger and stronger but at some point of time generator becomes stronger than discriminator based on the feedback provided by the discriminator and starts generating images so close to that of real images that demarcation between images of fake nature from those that are real becomes an unattainable task. It is the time when $p(x|y)$ for discriminator becomes 0.5 (that is it randomly classifies images as real and fake). It is important to stop the training at that point because then the feedback from the discriminator is of no use and it may lead to deteriorate our model. Fig. 3.1 represents the architecture of GAN.

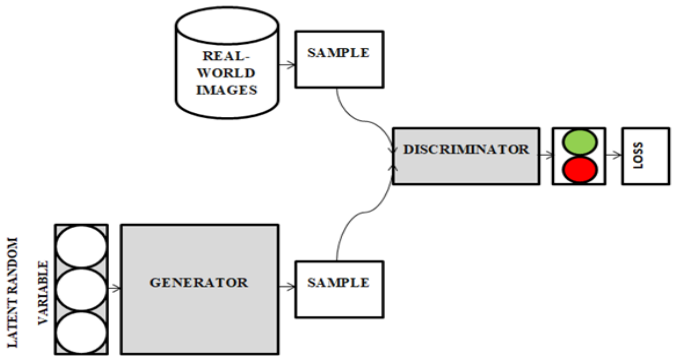

Figure 3.1: Architecture of GAN

This process is called Adversarial Training.

$Z \rightarrow$ Noise vector

$G(z) \rightarrow$ Generator's output $\rightarrow$ x_{fake}

$x \rightarrow$ Training Sample : x_{real}

$D(x) \rightarrow$ Discriminator's output for $x_{real} \rightarrow P(y \mid x_{real}) \rightarrow \{0,1\}$

$D(G(z)) \rightarrow$ Discriminator's output for $x_{fake} \rightarrow P(y \mid x_{fake}) \rightarrow \{0,1\}$

At Discriminator D,

D(x) -> should be maximized

D(G(z)) -> should be minimized

At Generator G,

D(G(z)) -> should be maximized

Mathematically,

At Discriminator D,

$$Dloss_{real} = \log(D(x)) \tag{3.1}$$

$$Dloss_{fake} = \log\left(1 - D\left(G(G(z))\right)\right) \tag{3.2}$$

$$Dloss = Dloss_{real} + Dloss_{fake} = \log(D(x)) + \log\left(1 - D\left(G(G(z))\right)\right) \tag{3.3}$$

$$\text{Total cost} = \frac{1}{m}\sum_{i=1}^{m} \log(x^i) + \log\left(1 - D\left(G(z^i)\right)\right) \tag{3.4}$$

Here, the equation 3.3 represents loss function for Discriminator where D(x) is maximized whereas D(G(x)) is minimized. Equation 3.4 and Equation 3.6 represent the actual cost function which will be minimized while training. Similar to the discriminator, Equation 3.5 represents generator loss but contrary to discriminator, here D(G(z)) is maximized.

At Generator G,

$$\text{Gloss} = \log\left(1 - D\left(G(G(z))\right)\right) \text{ or } -\log\left(D(G(z))\right) \tag{3.5}$$

$$\text{Total cost} = \frac{1}{m}\sum_{i=1}^{m} \log\left(1 - D\left(G(z^i)\right)\right) \tag{3.6}$$

$$\min_G \max_D V(D, G) = E_{x \sim p_{data(x)}}\left[\log D(x)\right] + E_{x \sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right] \tag{3.7}$$

$$\max_D V(D) = \underbrace{E_{x \sim p_{data}(x)}\left[\log D(x)\right]}_{\text{recognize real images better}} + \underbrace{E_{x \sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]}_{\text{recognize generated images better}} \tag{3.8}$$

$$\min_G V(G) = \underbrace{E_{x \sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]}_{\text{Optimize G that can fool the discriminator the most}} \tag{3.9}$$

So in terms of game theory discriminator and generator play the minimax game where value function is V(D,G) where generator tries to minimise the value of function given in Equation 3.9 while discriminator tries to maximize this value as shown in equation 3.8. The corresponding minimax equation is given by Equation 3.7.

There are various types of GAN based on their architecture:

- Deep Convolutional GANs (DCGANs)
- Cycle GAN
- Vanilla GAN
- Conditional GAN(C-GAN)
- Discover Cross-Domain Relations with Generative Adversarial Networks(Disco GANS)

3.4 CONDITIONAL GAN

Until now the working of the GANs was unsupervised and there was no way you could generate data instances according to your own will. GAN will be of little use if we are not able to generate objects according to our input and to solve that problem Conditional GAN (or C-GAN) were introduced.

In this we apply a condition on both generator and discriminator this could be image class or some other property which controls the output and instructs the generator about what to do.

These kinds of GANs are constructed by providing a generator and discriminator, a feature vector which is derived from the features of the image that encode a class (like type of hair for classes of man or woman. The discriminator not only judges the generator on the basis of distinction between fake and real images but also how much the generated fake image matches with the label it is referred to.

3.4.1 Architecture of Conditional GAN

The architecture of C-GAN is similar to that of DCGAN with just the difference that C-GAN has one additional one hot encoded input layer which is used to control the output.

A simple C-GAN architecture can be represented as in Fig 3.2:

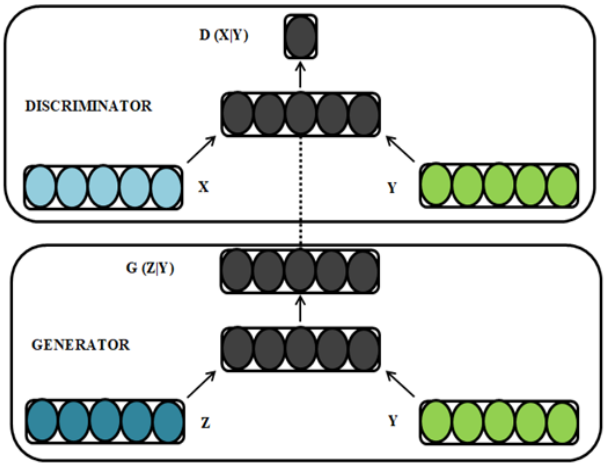

Figure 3.2: Architecture of C-GAN

Here first the input noise vector is concatenated with the feature vector which is passed through the dense and convolution layer of the generator and generates the image. Then this generated image is passed as input to the discriminator alongside the real image and feature vector. Discriminator concatenates the feature vector with both fake and real image and then again passes it through a convolution network to

ascertain the probability of an image being of real/fake nature. It then provides feedback to the generator via back propagation.

3.5 EMBEDDING

3.5.1 Word Embedding

We can encode the text by generating a sparse matrix for the whole vocabulary where each entry corresponds to a word and we can represent a word by an array with all 0s and 1 at a unique position describing the word. But this thing which works well for digits is not suitable with words because this method does not distinguish between related and unrelated words therefore we use a different method for words which also represent a word meaning a numeric vector but now two similar words have similar corresponding vectors. It is possible to do analogies like king is to man is same as queen is to woman. There are different methods for word embedding like Word2vec, GloVe, and FastText etc. Fig. 3.3 represents a visualization of word embeddings.

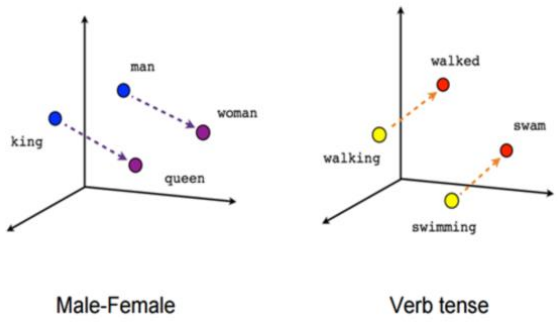

Figure 3.3: Word embedding visualisation

3.5.2 Sentence Embedding

Many times we need to take a step ahead and cypher meaning of the whole sentence. This is known as sentence embedding. It allows us to not calculate individuals word

embedding but rather see sentences as a whole. It is a very widely used technique for sentiment analysis or predicting the values. Calculating centroid distance was once a widely used technique for sentence embedding but its limitation to distinguish between the sequence of two sentences has made it very less popular.

Sentence-BERT is one of the most widely used sentence embedding techniques nowadays.

3.5.2.1 Sentence BERT

Sentence BERT is very good at making embeddings because of its adaptability. The word embeddings change dynamically on the basis of the vector surrounding them thus two sentences I ate an apple and Apple bought a startup have completely different embeddings because of the different context. It gets trained using a sentence with added noise where it aims to bring back the sentence into its initial form

This training is done on GBs of data from various sources. To give good results fine tuning is applied on BERT which means that we reward the model with sentence embedding having the property that cosine similarity for pairs of sentences represents semantic similarity for the sentences. If we find such a model then we can generate sentence embedding once and then find the cosine similarity for each pair. This can be done with the help of Siamese Neural Network where we obtain embedding for pairs of sentences and then concatenate them. Fig. 3.4 represents SBERT trained for classification objective.

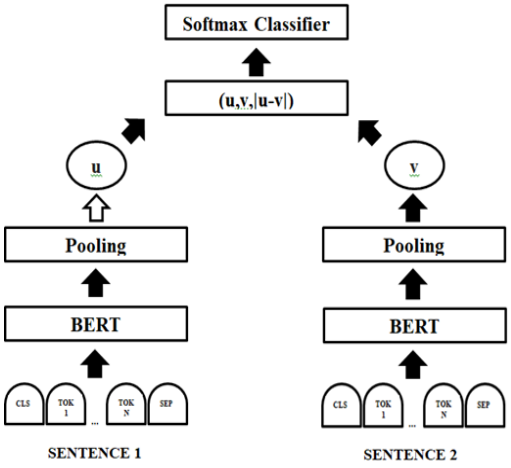

Figure 3.4: Siamese neural network in Sentence BERT

CHAPTER 4

PROPOSED METHODOLOGY

The following chapter aims to explain the work done in detail. It explains how a two-staged StackGAN architecture has been used to generate highly realistic images from textual description. Section 4.1 explains how relevant data for our problem domain has been created. Section 4.2 represents our model architecture and its major components. This also contains implementation details of Stage-I and Stage-II GANs, discussing the loss functions associated as well as their corresponding optimizers.

4.1 DATASET AND STRUCTURE OF DATA

We used CelebFaces Attributes Dataset (CelebA) for generating textual descriptions required for training our model. It is a large-scale dataset with facial attributes of around 2,02,599 celebrity images each covering large pose variations and diversities. These cover around 10,177 identities alongside five landmark places and labels comprising of 40 features per image. This aforementioned dataset commonly acts as training & testing subsets, to be employed in a plethora of tasks in the field of Computer Vision.

In our work, we categorized the 40 attributes list in the CelebA dataset corresponding to each image into six different groups, each such group describing some peculiar facial characteristic. These groups were created based on the facial shape, hairstyle, appearance-describing attributes (such as young, pale etc.) accessories worn, hairstyle of the person and other facial features such as eyes, nose, lips etc. The attribute representing gender is used to identify a person as male/female, indicating whether to use ‘he’ or ‘she’ for the person in order to generate textual descriptions. Further, a dictionary is created with keys representing attributes from CelebA dataset and values from the text with which we want to replace them in the descriptions. While creating a sentence, we first appended a prefix corresponding to the unique group (Eg: “He has an ” for describing oval face attribute) and then we added the corresponding value from the dictionary created. After summing up all these individual sentences about an individual, we finally obtained the entire textual description highlighting the person’s facial features, face outline as well as relevant

accessories based on which we generated images. Table 4.1 presents the description of the dataset used.

Table 4.1: Description of dataset

Face images of various celebrities	202599
Unique identities	10177
Binary attribute annotations per image	40
Landmark locations	5

4.2 NETWORK ARCHITECTURE

Our model is based on sketch-refinement methodology for which we implemented a two staged GAN architecture with StackGAN, capable of converting the arduous task into relatively more feasible sub-tasks. Here, the Stage-I GAN makes use of the annotations to sketch the basic object colors and shapes. Consequently, the images generated are of poor resolution. Then, by means of the outputs obtained from Stage-I serving as inputs alongside the annotations used, Stage-II GAN produces detailed, high quality, realistic images of significant resolution. This is possible courtesy of the ability of the Stage-II GAN to improve upon the issues in images of Stage-I GAN, whilst incorporating finer details during the process of refinement.

This way, we can finally obtain desired high-resolution images but subsequent sections explain each component of our architecture in detail. The major components of both the GANs are textual embedding encoder, embedding compressor network, Conditioning Augmentation network, Generator as well as Discriminator.

The subsequent sections explain each of these in detail. In each of the stages, C-GANS are responsible for construction of the network of generators. Using them, it is possible to create fake but photo-realistic looking images after training on an image data as they are used to learn a conditional density model. Fig. 4.1 represents the architecture of the proposed model.

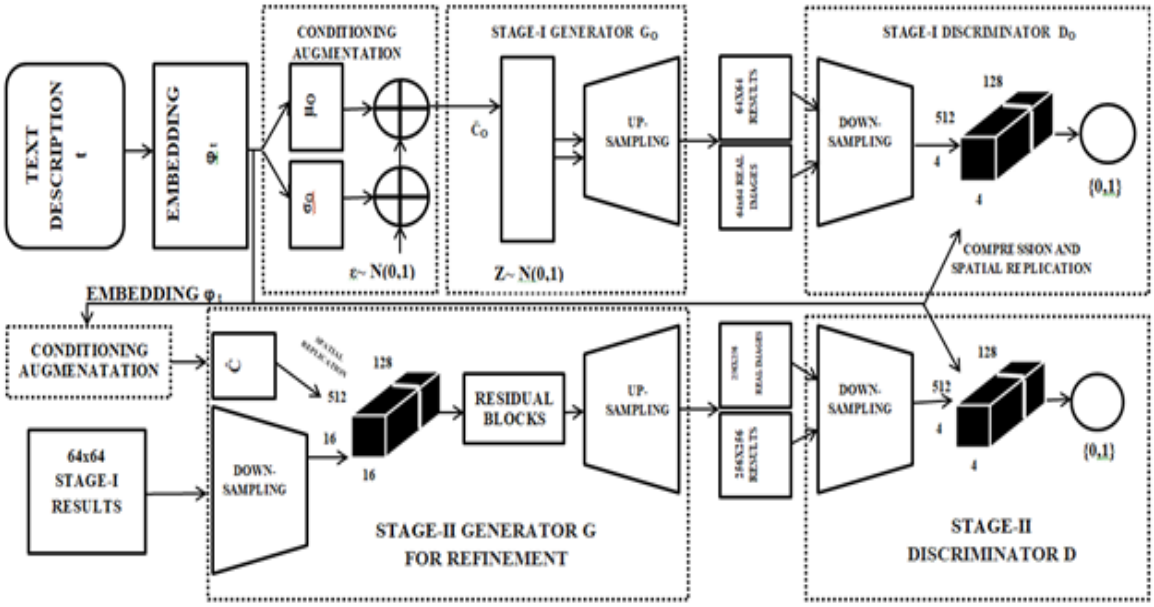

Figure 4.1: Architecture of proposed model

4.2.1 Textual embedding encoder network

The first step as shown in the Fig. 4.1 was to encode the text description into an embedding for which we have used a pre-trained Sentence-BERT network Fig. 3.4. It introduces pooling to the token embeddings generated by BERT in order for creating a fixed size sentence embedding. These can be used for unsupervised tasks to encode the semantics of sentence data. The major purpose of creating embeddings is to create vectors of constant size from input text of random size. The text encoder network encodes a sentence to a 1,024 dimensional text embedding. This network is common to both of the stages. Table 4.2 describes the shape of the training and test set embeddings.

Table 4.2: Description of the shape of embeddings

Train data embeddings shape	(151949,10,1024)
Test data embeddings shape	(50650,10,1024)

4.2.2 Conditioning augmentation network

Large number of dimensions is a common feature of text-conditioned latent space that can lead to discontinuities. Hence, we have utilized Conditioning Augmentation technique for extraction of latent variables from text embeddings in order to facilitate creation of supplementary variables for conditioning. In this technique, the conditioning augmentation block represents a single linear layer in which the embeddings of shape (1024,) are fed and in return, it provides a tensor of shape (256,) which is considered as the relevant input to be understood by GAN. This network is used to sample random latent variables from a distribution given by:

$$N\left(\mu_0\left(\varphi_t\right), \sum_0\left(\varphi_t\right)\right)$$

.

This technique adds more randomness in the network. Also, helps in making the generator more robust by capturing objects with much more diversity in terms of poses and appearances. With a higher number of image-text pairs, we can train a robust network that can handle perturbations. Through this, we not only just considered the training images and the noise inspiration but we also considered some additional information represented as c which in our case was the phrase used to describe an image. This information is used by discriminator to check if the phrase matches with the image and for the generator, it is used as a part of inspiration for creating the image.

After obtaining textual embeddings from Sentence-Bert, a fully-connected layer is provided with them so as to generate values such as the mean as well as standard deviation. These are then used to create a diagonal covariance matrix by placing in a diagonal of the matrix. Finally, we create a Gaussian distribution using these which can be represented as follows:

$$N\left(\mu_0\left(\varphi_t\right), \sum_0\left(\varphi_t\right)\right)$$

Then, we sampled from the Gaussian distribution that we just created. To sample $\hat{C}_0$ we first take the element-wise multiplication of standard deviation and then add the output to mean. The formula to compute is as follows:

$$\hat{C}_0 = \mu_0 + \sigma_0 \; e \; N(0, I) \tag{4.1}$$

4.2.3 Stage-I GAN

We have proposed a Conditional GAN architecture for Stage-I and further explanation regarding Generator and Discriminator networks are provided in the following sections.

4.2.3.1 Generator

An architecture of Deep Convolutional Neural Network with several 2D up-sampling blocks, each containing an up-sampling and convolutional layer along with a batch normalization layer, has been created by us as is shown in the diagram Fig. 4.1. The generator network is a Conditional GAN, which is conditioned on the conditioning variable c and the random variable z sampled from a Gaussian distribution with dimension N. After concatenating the text-conditioning variable with the noise variable, we created a dense layer containing 128*8*4*4 (16,384) nodes - two-dimensional tensor into four-dimensional tensor

It further generates a low-resolution image of dimensions 64x64x3 which might represent just facial outline or primitive facial features with a lot of defects. The architecture implemented in our work can be seen in Table 4.3 which contains several up-sampling blocks made from several convolutional layers followed by batch normalization or activation layers.

Table 4.3: Architecture of Stage-I Generator

Layer (type)	**Output Shape**	**Connected to**
input_1 (Input Layer)	(None, 1024)	

dense_1 (Dense)	(None, 256)	input_1[0][0]
leakyReLU_1 (LeakyReLU)	(None, 256)	dense_1[0][0]
lambda_1 (Lambda)	(None, 128)	leakyReLU_1[0][0]
input_2 (Input Layer)	(None, 100)	
concatenate_1 (Concatenate)	(None, 228)	lambda_1[0][0] input_2[0][0]
dense_2 (Dense)	(None, 16384)	concatenate_1[0][0]
reLU (ReLU) + reshape (Reshape) [4 Blocks]	(None, 4, 4, 1024) → (None, 32, 32, 128)	dense_2[0][0]
upsampling_2D (UpSampling 2D) [4 Blocks]	(None, 8, 8, 1024) → (None, 64, 64, 128)	reLU[0][0]
conv2D (Conv2D) [4 Blocks]	(None, 8, 8, 512) → (None, 64, 64, 64)	upsampling_2D[0][0]
batch_normalization (BatchNorm) [4 Blocks]	(None, 8, 8, 512) → (None, 64, 64, 64)	conv2D[0][0]
reLU_6 (ReLU)	(None, 64, 64, 64)	batch_normalization[0][0]
conv2D_6 (Conv2D)	(None, 64, 64, 3)	reLU_6[0][0]
activation_1 (Activation)	(None, 64, 64, 3)	conv2D_6[0][0]

Total parameters: 10,270,400

Trainable parameters: 10,268,480

Non-trainable parameters: 1,920

4.2.3.2 Discriminator

Another Deep Convolutional Neural Network, consisting of various downsampling blocks as convolutional layers, that has been used by us is the Discriminator. These layers create feature maps which are further concatenated to textual embeddings. The text embeddings are compressed to less dimensions (three-dimensional tensor) and the image is generated via a number of down-sampling blocks, till the time it becomes a two-dimensional tensor. It is then concatenated and fed to a convolutional layer to facilitate joint learning of features vis-a-vis the textual description and image. Ultimately, we ascertained the probability of discerning actual data distribution images from those obtained via generators with the help of a fully connected layer containing a single node.

Hence, after stage-I, we obtained 64*64 low resolution images with various distortions and vagueness. Table 4.4 describes the architecture of Stage-I discriminator.

Table 4.4: Architecture of Stage-I Discriminator

Layer (type)	Output Shape	Connected to
input_1 (Input Layer)	(None, 64, 64, 3)	
conv2D_1 (Conv2D)	(None, 32, 32, 64)	input_1[0][0]
leakyReLU (LeakyReLU) [3 Blocks]	(None, 32, 32, 64) → (None, 8, 8, 256)	dense_1[0][0]
conv2D (Conv2D) [3 Blocks]	(None,16, 16, 128) → (None, 4, 4, 512)	leakyReLU[0][0]
batch_normalization (BatchNorm) [3 Blocks]	(None, 16, 16, 128) → (None, 4, 4, 512)	conv2D[0][0]

leakyReLU_2 (LeakyReLU)	(None,4, 4, 512)	batch_normalization[0][0]
input_2 (Input Layer)	(None, 4, 4, 128)	
concatenate_1 (Concatenate)	(None, 64, 64, 3)	leakyReLU_2[0][0] input_2[0][0]
conv2D_2 (Conv2D)	(None,4, 4, 512)	concatenate_1[0][0]
Batch_normalization_2 (BatchNorm)	(None,4, 4, 512)	conv2D_2[0][0]
leakyReLU_3 (LeakyReLU)	(None,4, 4, 512)	Batch_normalization_2[0][0]
flatten_1 (Flatten)	(None, 8192)	leakyReLU_3[0][0]
dense_1 (Dense)	(None, 1)	flatten_1[0][0]
activation_1 (Activation)	(None, 1)	dense_1[0][0]

Total parameters: 3,097,601

Trainable parameters: 3,094,785

Non-trainable parameters: 2,816

Table 4.5 provides the values of hyperparameters for Stage-I GAN.

Table 4.5: Hyperparameters of Stage-I GAN

Training Stage-I GAN

Image size	64
Batch size	64
z_dim	100

generator_lr	0.0002
discriminator_lr	0.0002
stage1_lr_decay_step	600
Epochs	1000
conditional_dim	128

4.2.4 Stage-II GAN

The generator network used in this stage employs a framework of encoder and decoder. It is assumed that randomness has been encountered by images generated from Stage-I GAN. Therefore, in this stage, random noise z is not taken into consideration. For the purpose of production of various means and SD, the Conditioning Augmentations of the two stages have separate fully-connected layers. Hence, even the information present in Stage-I GAN image in morphed form can be obtained with relative ease in Stage-II GAN. Poor resolution image generated by the Stack-I GAN as well as the textual descriptions help condition the Stack-II GAN and consequently high resolution images that consider more photorealistic details are yielded.

4.2.4.1 Generator

In this deep convolutional neural network, we have used lower dimensional Gaussian Conditioning Vector which is later transformed into a 3D tensor and the sample generated from the last stage is further compressed to 2D tensor by various downsampling blocks which generated tensors with shape 16x16x512 as can be seen from the given architecture to generate image features. Both of these are further concatenated into a single tensor with shape (batch_size, 640) and finally, provided to the generator to encode the textual features alongside the image. The output of this is further fed through a series of

upsampling blocks that ultimately generated a high-resolution, more photorealistic image having dimensions 256x256x3.

4.2.4.2 Discriminator

Structure of the discriminator is similar to the one used in Stage-I GAN in the sense that it still employs a deep convolutional neural network, albeit with an increased number of downsampling blocks as the input size in this stage is larger. These blocks are further followed by a concatenation block and then a classifier. This architecture helps obtain arrangement of better nature between the conditioning text and the image that has been input. By considering real images alongside their corresponding textual descriptions as positive sample pairs and real images with text embeddings that have been mismatched, fake with embeddings of text as negative sample pairs, the Discriminator works.

Table 4.6 provides the values of hyperparameters for Stage-II GAN.

Table 4.6: Hyperparameters of Stage-II GAN

Training Stage-II GAN

hr_mage_size	(256,256)
lr_mage_size	(64,64)
batch_size	8
z_dim	100
stage2_generator_lr	0.0002
stage2_discriminator_lr	0.0002
stage2_learning_rate_decay_step	600
Epochs	10

conditional_dim	128

4.2.5 Adam optimizer

Adam Optimisation is a very popular and effective optimisation technique that is employed as an alternative to traditional gradient descent procedures for updation of weights in a network on the basis of training data. The stochastic gradient descent maintains similar learning rates for all weight but in this each weight adapts its own learning rate as the learning continues. This technique ascertains an exponential shifting average of the gradient along with the squared gradient. Control of the decay rates of shifting moving averages is done by parameters beta1 and beta2. We have used the Adam optimizer with the learning rate=2 x 10-3 and the beta_1 value=0.5 and beta_2= 999 x 10-3.

4.2.6 Loss functions

Kullback-Leibler divergence (KL divergence) loss function also known as relative entropy function originated from information theory to measure the information in the data is a very important loss function which tells us how a probability distribution is different from other reference probability distribution. For two identical distributions the value of KL divergence loss is 0. The notation for KL loss function is KL(A||B) where || is the divergence operator. It is calculated as the negative sum of probability of each event in A multiplied by the log of the probability of event B over the probability of event A. The KL Divergence loss function formula is given below as Equation 4.2:

$$KL(A \mid\mid B) = -\ \text{sum x in X } A(x) * \log(B(x) / A(x)). \quad (4.2)$$

Similar to any other GAN, training of the generator and discriminator networks in Stack-I and Stage-II GAN is done by minimizing generator network loss and maximizing

discriminator network loss. KL_loss is a custom loss function, specified in the Training of generators of both the GANs.

4.2.6.1 Stage – I

4.2.6.1.1 Generator

Equation 4.3 represents the loss function for the generator network in Stage-I GAN, in which both networks are conditioned on the text embeddings. Also, it includes a KL-divergence term to the loss function.

$$L_{G_0} = E_{z\sim p_z,\, t\sim p_{data}}\left[\log\left(1 - D_0\left(G_0\left(z, \hat{c}_0\right), \varphi_t\right)\right)\right] + \lambda D_{KL}\left(N\left(\mu_0\left(\varphi_t\right), \sum_0\left(\varphi_t\right)\right) \| N(0, I)\right) \tag{4.3}$$

4.2.6.1.2 Discriminator

Equation 4.4 represents the loss function for the discriminator network, in which both networks are conditioned on the text embeddings.

$$L_{D_0} = E_{(I_0,\, t)\sim p_{data}}\left[\log D_0\left(I_0, \varphi_t\right)\right] + E_{z\sim p_z,\, t\sim p_{data}}\left[\log\left(1 - D_0\left(G_0\left(z, \hat{c}_0\right), \varphi_t\right)\right)\right] \tag{4.4}$$

KL_loss is a custom loss function, specified in the Training of Stage-I GAN.

4.2.6.2 Stage – II

4.2.6.2.1 Generator

Equation 4.5 here represents the loss function for the generator network in Stage-II GAN. This network also consists of a Kullback-Leibler (KL) divergence term to its loss function.

$$L_G = E_{s_0\sim p_{G_0},\, t\sim p_{data}}\left[\log\left(1 - D\left(G\left(s_0, \hat{c}\right), \varphi_t\right)\right)\right] + \lambda D_{KL}\left(N\left(\mu\left(\varphi_t\right), \sum\left(\varphi_t\right)\right) \| N(0, I)\right) \tag{4.5}$$

4.2.6.2.2 Discriminator

Equation 4.6 represents the loss function for the discriminator network, in which both networks are conditioned on the text embeddings. One major difference is that the generator network has x and c as inputs, where x is the image generated by the Stage-I and c is the Conditioning Augmentation variable.

$$L_D = E_{(I, t)\sim p_{data}}\left[\log D\left(I, \varphi_t\right)\right] + E_{s_0\sim pG_0,\, t\sim p_{data}}\left[\log\left(1 - D\left(G\left(s_0, \hat{c}\right), \varphi_t\right)\right)\right] \tag{4.6}$$

CHAPTER 5

RESULTS AND EXPERIMENTATION

The following chapter presents the results obtained from the conducted experiments. Section 5.1 discusses Inception Score metric. Section 5.2 and Section 5.3 depict the variation of discriminator and generator loss respectively with the number of training epochs. Section 5.4 provides a visual representation of the results obtained.

5.1 INCEPTION SCORE

The inception score (in short IS) with its origin from Inception Classifier is a very widely used metric for judging the performance of images generated by General Adversarial Network. Inception score outputs a floating point number which is directly proportional to how realistic are the images generated by the GANs. The upper bound of score is infinity but in practicality there usually emerges a non infinite ceiling. It gets well correlated with the human evaluation and can act as an alternative to a human actually judging the images for the quality. It takes into account both factors that images have variety and each image actually looks like a data instance. To score to be high both the conditions must be satisfied.

To calculate both the conditions, the Inception score makes use of the Inception Classifier which classifies the image to a particular label by providing the probability values for each label. Now as similar labels sum to give focussed distribution while different labels sum to a uniform distribution so we can use this classifier on generated images from GAN by passing the fake images through the classifier which can predict the label of images in the output. So the score combines two factors that each image must be distinct (so probability distribution for a single picture must be narrow i.e focussed on one peak) and collectively it must be uniform(for the sum the distribution must be uniform) telling that the collection has variety. We also used this score for the purpose of checking how our

model works and we obtained IS of 4.04 ± 0.05 over 10 epochs. Our inception score indicates that for every image x, textual encoding z and class label y, the value of marginal distribution p(y) given by:

$$p(y) = \int_x p(y|x = G(z))dz \tag{5.1}$$

It possesses entropyvalue that is large in magnitude and of a nature comparable to p(y|x).

5.2 DISCRIMINATOR LOSS

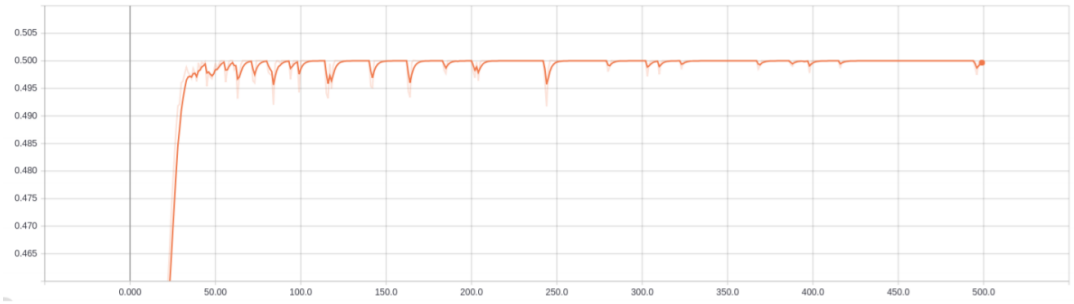

Figure 5.1: Variation of Discriminator Loss with the Number of Training Epochs

5.3 GENERATOR LOSS

Figure 5.2: Variation of Generator Loss with the Number of Training Epochs

The plots obtained from Stage-II can be shown in Fig. 5.1 and Fig. 5.2. These help in determining when to stop the training of GANs. If losses are not decreasing further, it indicates that the training can be stopped as there is no chance of improvement or if losses keep on increasing, then it indicates that the training must be stopped. But, if the losses are gradually decreasing, then we must continue training.

5.4 VISUAL RESULTS

Fig. 5.3 and 5.4 represent the visual results of Stage-I and Stage-II respectively.

5.4.1 Stage-I

Figure 5.3: Visual results of Stage-I

5.4.2 Stage-II

The man has an oval face with high cheekbones. He has wavy hair which is brown colored. He has a marginally open mouth. The youthful appealing man is smiling.

The lady has high cheekbones. She has straight hair which is brown colored. She has arched eyebrows and a somewhat open mouth. The youthful appealing lady has makeup. She is wearing lipstick.

The man looks youthful and his hair is brown colored.

The lady has an oval face. She has straight hair which is brown colored. The youthful appealing lady has makeup. She is wearing lipstick.

The lady has an oval face with high cheekbones. She has enormous lips and small eyes with curved eyebrows and a somewhat open mouth. She has straight hair which is brown colored. She is wearing earrings as well as lipstick. The youthful appealing lady has makeup.

Figure 5.4: Visual results of Stage-II

CHAPTER 6

CONCLUSION AND FUTURE WORK

The following chapter presents the conclusion arrived at from the conducted research along with prospective directions of future work.

The Stack GAN architecture implemented in this report has shown promising results on our dataset. It is based on the sketch-refinement technique. In short, we obtained IS of 4.04 ± 0.05 over 10 epochs and we successfully synthesized 256*256 photorealistic images. Stage-I GAN takes a sentence as input and outputs a resultant picture containing colors and shapes of rudimentary nature whereas the second one (Stage-II) takes as input an image of lower resolution from Stage-I along with the initial sentence. This way it is able to synthesize an image possessing a resolution larger in magnitude having fine-tuned the intricacies (264*264). We have successfully synthesized people's pictures based on their description.

A lot of research is still going on in the field of GANs to handle the existing challenges such as Mode Collapse and Failure to converge. In future, we will be proposing an appropriate network in order to solve these. Apart from this, we will be increasing the size of our dataset further with a greater variety of textual descriptions which will enhance the chances of better training. Further, we are also proposing capsule networks over CNN architectures due to their obvious advantages. We also propose to figure out a better evaluation metric other than inception score so that we can capture the similarity without using classes.

REFERENCES

[1] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal deep learning,” 2011.

[2] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” 2016.

[3] H. Zhang *et al.*, “StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks,” 2017, doi: 10.1109/ICCV.2017.629.

[4] Z. Zhang, Y. Xie, and L. Yang, “Photographic Text-to-Image Synthesis with a Hierarchically-Nested Adversarial Network,” 2018, doi: 10.1109/CVPR.2018.00649.

[5] T. Xu *et al.*, “AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks,” 2018, doi: 10.1109/CVPR.2018.00143.

[6] A. Gatt *et al.*, “Face2Text: Collecting an annotated image description corpus for the generation of rich face descriptions,” 2019.

[7] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments,” 2008, Accessed: Dec. 06, 2020. [Online]. Available: http://vis-www.cs.umass.edu/lfw/.

[8] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, “MS-celeb-1M: A dataset and benchmark for large-scale face recognition,” 2016, doi: 10.1007/978-3-319-46487-9_6.

[9] I. J. Goodfellow *et al.*, “Generative adversarial nets,” 2014, doi: 10.3156/jsoft.29.5_177_2.

[10] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2016.

[11] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *IEEE Trans. Neural Networks*, 1997,

doi: 10.1109/72.554195.

[12] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” 2014.

[13] M. Li, W. Zuo, and D. Zhang, “Convolutional Network for Attribute-driven and Identity-preserving Human Face Generation,” 2016.

[14] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, “Deep convolutional inverse graphics network,” 2015.

[15] R. Huang, S. Zhang, T. Li, and R. He, “Beyond Face Rotation: Global and Local Perception GAN for Photorealistic and Identity Preserving Frontal View Synthesis,” 2017, doi: 10.1109/ICCV.2017.267.

[16] L. Tran, X. Yin, and X. Liu, “Disentangled representation learning GAN for pose-invariant face recognition,” 2017, doi: 10.1109/CVPR.2017.141.

[17] J. Zhao *et al.*, “Towards Pose Invariant Face Recognition in the Wild,” 2018, doi: 10.1109/CVPR.2018.00235.

[18] M. Z. Khan *et al.*, “A Realistic Image Generation of Face from Text Description using the Fully Trained Generative Adversarial Networks,” *IEEE Access*, pp. 1–1, Aug. 2020, doi: 10.1109/access.2020.3015656.

[19] Y. Shen, P. Luo, J. Yan, X. Wang, and X. Tang, “FaceID-GAN: Learning a Symmetry Three-Player GAN for Identity-Preserving Face Synthesis,” 2018, doi: 10.1109/CVPR.2018.00092.

[20] Y. Shen, B. Zhou, P. Luo, and X. Tang, “FaceFeat-GAN: a Two-stage approach for identity-preserving face synthesis,” *arXiv*. 2018.