Face IT

Submitted in partial fulfilment of the requirements
of the degree of

Bachelor of Engineering

by

Monil Diwan, 60003160014
Karan Doshi, 60003160015
Moxa Doshi, 60003160016

Project Guide
Prof. Purva Raut

Department of Information Technology
D. J. Sanghvi College of Engineering,
Mumbai – 400 056
2019-20

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

CERTIFICATE

This is to certify that the project entitled **FACE IT** is a bonafide work of **Monil Diwan (60003160014)**, **Karan Doshi (60003160015)** and **Moxa Doshi (60003160016)** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**.

Guide
(Prof. Purva Raut)

Dr. Vinaya Sawant
Head of Department

Dr. Hari Vasudevan
Principal

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Project Report Approval for B. E.

This project report entitled ***Face IT*** by ***Monil Diwan, Karan Doshi & Moxa Doshi*** is approved for the degree of Information Technology.

Examiners 1.--

2.--

Date:

Place:

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

--	--
Monil Diwan (60003160014)	
--	--
Karan Doshi (60003160015)	
--	--
Moxa Doshi (60003160016)	
--	--

Date:

Acknowledgement

This project report consumed huge amount of work, research, passion and dedication. For completing this report, we had to take help and guidance of some respected people, who deserve our greatest gratitude. The completion of this report gives us much pleasure. We owe a profound gratitude to our project guide, Prof. Purva Raut, who not only introduced us to the methodology of work but also took a keen interest in our report and guided us all along. In addition, we would also like to thank our respected Principal, Dr. Hari Vasudevan, the Head of Department, Dr. Meera Narvekar, and Prof. A.R. Joshi for giving us the support and guidance to work on this report. We are also thankful to the college for providing us with the necessary resources and also many thanks to the staff of college for their valuable co-operation.

Many people, especially our classmates and team members itself, have made valuable suggestions which gave us an inspiration to improve our project. We thank them all for their help to complete the report.

Nevertheless, we express our gratitude towards the countless individuals who shared their knowledge and helped us in this report.

And finally, we are immensely grateful to all involved in the report as without their inspiration and immensely valuable suggestions it would not have been possible to develop the report within the prescribed time.

Monil Diwan
Karan Doshi
Moxa Doshi

Abstract

In this report we present a novel idea of hybrid algorithm a two stage Generative Adversarial Networks (GAN) each comprising of Convolutional Neural Network (CNN) tackling the problem of face inpainting. In order to address various peculiarities of face, we aid the user by accepting textual inputs for the same. We intend to achieve the results by applying pixel generation using GAN (Stage 1) combined with text to image conversion using GAN (Stage 2) benchmarking on a custom-made dataset for this task. The details of the implementation and the intermediate experiments along with the recorded observation, data curation and data pre-processing step are also presented.

List of Figures

List of Tables

Table of Contents

Page number

Chapter 1

Introduction

In this chapter we introduce the idea of our project, why we are doing it and how it will help the society. Various challenges are also discussed in this chapter.

1.1. Motivation/Objective

In crime investigation sector, recognition of the suspect is becoming difficult due to lack of proper pictorial evidence. At times it becomes difficult to recognize the person when only the partial face is available as evidence. In such cases there is absolutely no technology available to reconstruct the entire face. This drawback leads to misunderstanding among the investigation department because they may capture someone who is not guilty but just has similar partial facial feature as that of the suspect. In situations where the partial face is insufficient to generate the entire face, textual description can be very much useful as it will be more explicit in mentioning the facial features like eyes shape, hair color, skin color etc.

The objective of this project is to aid the crime investigation department by predicting criminal’s entire face just by a part of their face. The main audience of these software would be the crime investigation department.

1.2. Major Challenges

Different Individuals have different perception about a particular facial feature as each person's sense of understanding about a specific feature might be different. This may mislead the system to a considerable extent. This in turn will provide anomalous results and system will not perform as expected. For example, A crooked nose can be described as 'crooked' and 'Not very straight'. The above two phrases mean the same but are described in completely different words by different individuals.

For the system to predict the complete face from the given partial face, a minimum specific percentage of the face should be made available to system. Hence, for images where the available portion of the face does not cross the threshold, the system may not accurately predict the face because of lack of sufficient input data. When the input image is not bright enough for the system to detect a face in it, it will make the system unable to detect a person's complexion and hence hampering the output.

1.3. Report Overview

Introduction: The report consists of description of the system along with the objective behind the emergence of such an idea. The obstacles that were encountered by the system in achieving the expected result are also mentioned.

Literature survey: It includes an on-field interview of the immediate users of the system. To conclude the literature survey a concrete problem definition is made, scope of the system is identified and all the limitations as well as assumptions are listed.

Proposed Approach: Shortlisting, studying and combining multiple technologies boiled down to an approach towards building the system. Features of the system are described.

Project Management: A detailed architecture of the system is given to explain technology stack of the system. Use case diagram is provided to model the basic flow of the system and to provide specific need of the project. An activity diagram is given to show the flow of tasks and describe the entire process step by step.

Implementation: To implement the idea study of the existing system, their methodology or approaches, algorithms implemented and the technology used.

Testing: Various testing methods used for the project are explained in detail and the test cases are shown at every stage of the project.

Conclusion: The report also gives benefits/merits that the proposed system will provide to the society. Technical, operational and economic feasibility of the project is also studied.

Chapter 2

Literature Survey

In this chapter we have discussed all the papers, techniques, approaches we have studied for making this project. Detail information on every module's literature survey is given below.

2.1. Existing Work

Remarkable work in the chosen domain includes image competition of symmetric objects, text to image construction of flowers and birds, concept of stack GAN has been implemented. Data augmentation technique is also implemented individually.

2.1.1. Literature Related to Existing System

Pre-processing: Our project requires proper image with a face in it for training. Face detection in our project is implemented using YOLO algorithm. In [10] it detects faces present in the input images. YOLO is a latest algorithm which uses convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. This algorithm has provided exceptional results in detecting faces when a part of the face was masked.

Image completion: In [2] a generative image completion is used to implement face completion. The system directly generates the missing information is generated of the image based on neural network. The system ensures pixel faithfulness and local-global contents consistency. An incomplete image is given as input to the system which generates the whole image. The system fails to address unaligned images.

Another system [3] Deep Paint: A Tool for image inpainting focuses on filling the missing portion of an image that is either corrupted or unwanted. The applications of this system include: removing noise from an image, undoing deterioration (in historical or old photos), or to simply add or remove elements from an image.

Text to Image Synthesis: In [4] GAN text to image synthesis images of birds and flowers from the textual description provided by the user are generated. This system translates visual concepts from characters to pixels. This system focuses on translating text in the form of single-sentence human-written descriptions directly into image pixels.

In StackGAN, Text to photo realistic Image synthesis with Stacked Generative Adversarial Network. This system proposes a Stacked Generative Adversarial Networks (StackGAN) to generate 256x256 photo-realistic images conditioned on text descriptions. This system decomposes the bigger problem of generating pictures into smaller problem to generate high quality images.

2.1.1. Literature Related to Methodology/ Approach

Generative Image Completion: In [2], [3] and [4] system , deep generative model called Generative adversarial networks is used which consists of two parts viz. a generator which takes the incomplete image as the input and generates the whole image, and a discriminator which takes the generated image and the actual image as input for calculating the difference(Loss) between the two images and back-propagates the loss. Semantic regularization network is also used to maintain the harmony of the image. These model fails to generate the eye for an unaligned face. The proposed model fails to recover the correct color of the lip. Pixel-level recurrent neural network (PixelRNN) [] can be used to address this issue. Deep Paint: A tool for Image Inpainting [3] focuses on solving the inpainting problem using both Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) approaches. This system uses CNN with Sigmoid Euclidean Loss and a simplified PixelRNN.

Text to Image Synthesis: In this [4] and [5] system uses a novel deep architecture and GAN formulation to effectively bridge these advances in text and image modeling, translating visual concepts from characters to pixels.

This approach is to train a deep convolutional generative adversarial network (DC-GAN) conditioned on text features encoded by a hybrid character-level convolutional recurrent neural network. Both the generator network G and the discriminator network D perform feed-forward inference conditioned on the text feature. In [6] stackGAN the model uses two GAN networks. The Stage-I GAN sketches the primitive shape and colors of the object based on the given text description, yielding the Stage-I low-resolution images. The Stage-II GAN takes Stage-I results and text descriptions as inputs, and generates high-resolution images with photo-realistic details (256x256 size images).

2.1.3. Literature Related to Algorithm

Generative Image Completion: In [2] system proposes an effective object completion algorithm using a deep generative model. The input is first masked with noise pixels on randomly selected square region, and then fed into an autoencoder. While the encoder maps the masked input to hidden representations, the decoder generates a filled image as its output. The system then regularizes the training process of the generative model by introducing two adversarial losses: a local loss for the missing region to ensure the generated contents are semantically coherent, and a global one for the entire image to render more realistic and visually pleasing results.

GAN Text to Image Synthesis: Generative Adversarial Network is used which consists of a generator and a discriminator.

In [3] Deep Paint, the goal of inpainting is to fill in a portion of an image that is either corrupted or unwanted. The applications of in-painting include: removing noise from an image, undoing deterioration (in historical or old photos), or to simply add or remove elements from an image. Like a talented painter, these model selects the preferred architectures that would learn how to paint missing textures by looking at different areas of the image. For these the model uses a CNN neural network with a PixelRNN network.

2.1.4. Literature Related to Technology

Multiple technologies studies include:

1. CNN**:** In [2] Image inpainting using Convoluted Neural Networks and Recurrent Neural Networks. This technology uses blind and non-blind inpainting. In the blind image inpainting problem no information about the location of the regions that are corrupted or need to be filled in is given to the algorithm or network and the network must automatically locate as well as fill in these regions. On the other hand, in non-blind image inpainting the location of the pixels needed to be painted are known a priori and provided to the network in some form. This technology is capable of generating 32x32 images.

2. GAN: In [7] Image Inpainting using Generative Adversarial Networks. This technology uses 2 Neural Networks namely generator and discriminator collectively called as GAN. Generator generates images which are fed along with real images to the discriminator which will backpropagate the error to the generator. It is capable of generating 128x128 images.

3. STACK GAN: In [4] Text to Image Generation using StackGAN is showcased. Stacked Generative Adversarial Networks are used which interprets the input text and generates realistic images. Two stages of GANs are used which help in generating 256x256 images.

4. Data Augmentation technique. It generates multiple copies of the same image by flipping, shifting, rotating and scaling to provide different viewing angles of a particular image.

2.2. Observation on Existing Work

Image inpainting with Convoluted Neural Networks and Recurrent Neural Networks [3] predicts and generates images i.e. fills the missing part of the image from the given input of an image with superimposed noise. This technique though successfully generates the image but gives 32x32 resolution which are blur images. The objects need to be symmetric for this technology to work, it will fail for images which are not symmetric. This technology cannot handle unaligned images.

Image inpainting using Generative Adversarial Networks [2]. Because of using two neural networks, one for generator and the other for discriminator; this technique is able to generate comparatively clear images of 128x128 resolution. But the usage of two neural networks demands high computing power and the model for GAN needs a special highly configured GPU for training. This technique also cannot handle images if they are not aligned.

Stack GAN approach for Text to Image Synthesis [4]. Usage of multiple stages of GANs enables the system to generate 256x256 resolution images. But this approach makes the system highly resource intensive. The input data passes through series of GANs because of which the model learns it so well that it considers noise in the input dataset as parameters and learns it. This is known as overfitting problem which occurs due to the usage of multiple GANs.

Table 2.1: Comparison of Existing Works

Paper	Technique	Drawbacks
Image inpainting Using CNN/ RNN	Blind and non-blind inpainting. 32 x 32 images	Blur Images Cannot Handle Unaligned Faces
Image Inpainting using GAN	Uses single stage Generative Adversarial Networks	Cannot Handle Unaligned Faces.
Text to Image Generation	Stack GAN is used 256 x 256 Images	Prone to overfitting

Chapter 3

Proposed Approach

In this chapter, we define the exact problem definition and scope of the project along with certain primary assumptions and constraints.

3.1. Problem Definition

In crime investigation sector, recognition of the suspect is difficult due to lack of proper pictorial evidence. At times it becomes difficult to recognize the person when only the partial face is available as evidence. In such cases, there is absolutely no technology available to reconstruct the entire face. This drawback creates an obstruction in the investigation process. Through this project, we will try to provide a solution to this extremely critical problem. Partial part of the suspect, which is available as evidence would be fed into the system and the system will generate the entire face hence helping the investigation process. In this scenario, it is extremely difficult for an algorithm to predict various peculiarities of face like a mole, beard, eyes shape, etc. To solve this problem, we introduce an option of providing textual description of the suspect's image.

The description would be accepted from the user by high-level description for example 'french beard', 'wavy hair', etc. and after a dual combination of both the modules viz. face completion and text to feature conversion, a final output of the entire face is generated.

3.2. Scope

To achieve the above problem description, we use a technique called Image Inpainting. In this project we present a novel idea of hybrid algorithm a two stage Generative Adversarial Networks (GAN) each comprising of Convolutional Neural Network (CNN) tackling the problem of face inpainting.

Image inpainting is a technique which aims to fill the missing regions of an image with plausibly synthesized contents. Inpainting is achieved by Generative Adversarial Networks (GAN). GAN, as defined by Ian J. Goodfellow is a combination of two neural networks, Generator and Discriminator. We prolonged this idea by employing a generator network which takes masked image as input and inpaints the missing region. This image is then forwarded to the discriminator to distinguish it as real or fake and backpropagates the feedback to the generator. This process eventually allows generator to generate plausibly realistic images.

In order to address various peculiarities of face, we aid the user by accepting textual inputs for the same. We intend to achieve the results by applying pixel generation using GAN (Stage 1) combined with text to image conversion using GAN (Stage 2) benchmarking on a custom-made dataset for this task. The details of the implementation and the intermediate experiments along with the recorded observation, data curation and data pre-processing step are also presented.

3.2.1. Assumptions and Constraints

Assumptions:

- The image given as input by the user will contain a human face.
- The structure of the face in the input image is generic and is not differently featured.
- User of the system has active internet connection.

Constraints:

- Image given as input should be bright enough for the system to function.
- Individuals face with distorted structure will not be generated.
- Highly configured GPU is required to train the model.

So, we move forward with the described problem statement and considering the mentioned scope of the project.

Chapter 4

Project Management

In this chapter, we will focus on the detailed time management of the project. This chapter also gives an insight of feasibility of the project as well as explains project estimation metrics like COCOMO model and Function point analysis along with Risk Mitigation and Management.

4.1. Project Schedule

The project was efficiently scheduled for 1.5 years. Starting from January 2019 it extended till mid-2020. We will further see the exact timeline of the project proceedings.

4.1.1. Task Network Diagram

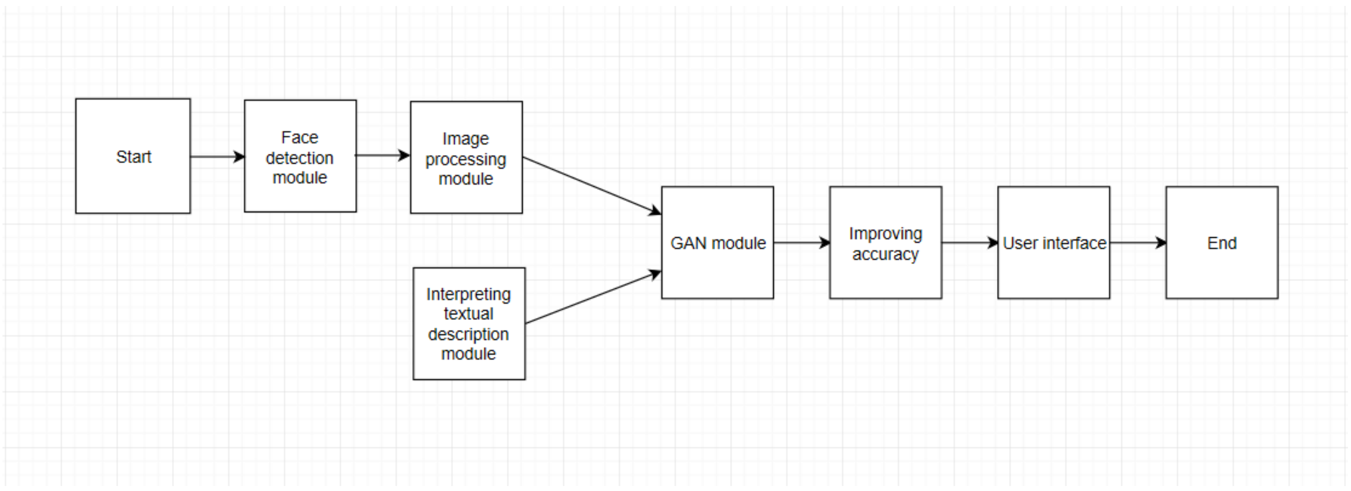


Figure 4.1. Task Network Diagram

4.1.2. Timeline Chart

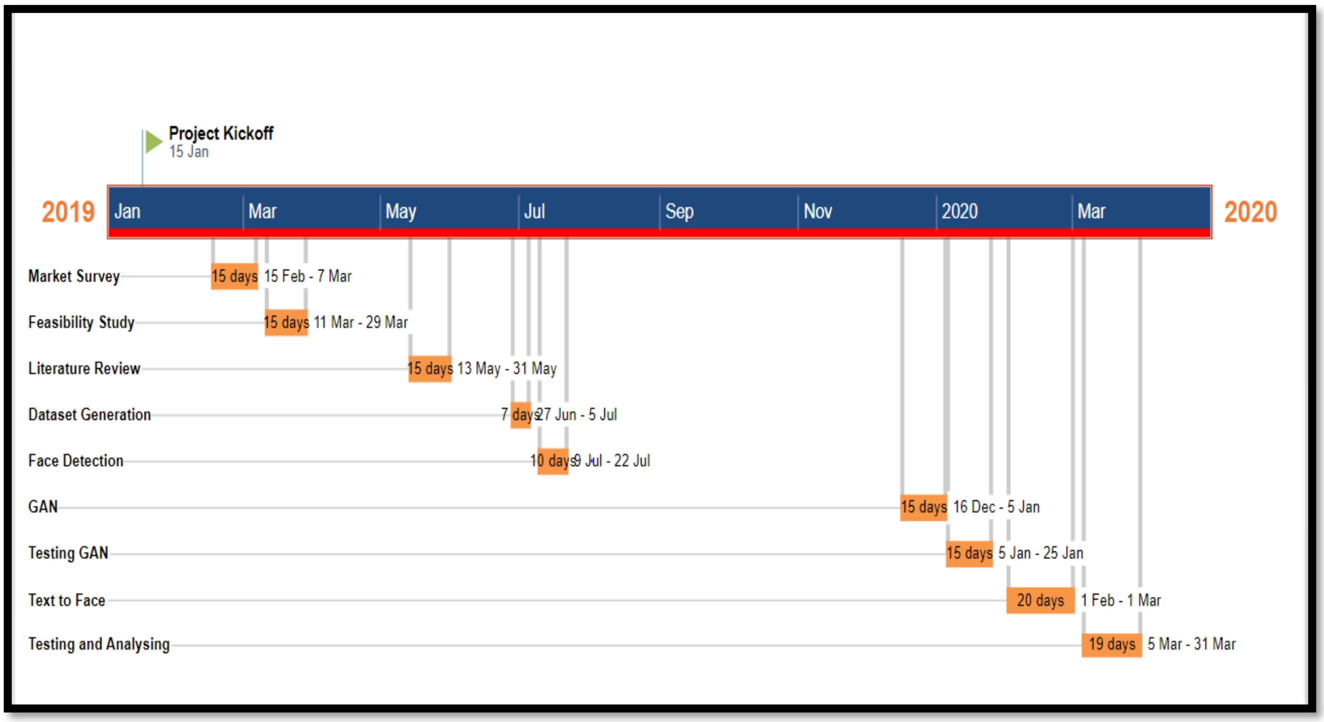


Figure 4.2. Timeline Chart

4.2. Feasibility Study

4.2.1. Technical Feasibility

To train the model, celeb A dataset is available which consists of center cropped faces of celebrities. To insert random noise in the picture, python Pillow library is available which can mask an image with black patches.

- Task performed by this system are well identified which includes face detection, image processing, data augmentation and face generation.
- Model used will be Generative Adversarial Networks and Natural language processing.
- Various Python Libraries are available such as TensorFlow and Keras for implementing Neural Networks which come built-in with Jupyter notebook and Google Collab.

4.2.2. Operational Feasibility

The system is extremely feasible considering the user's convenience. User only have to insert an image and its textual description in terms of input and he/she will receive 'top-k' suggestions from the system of the predicted face.

4.2.3. Economic Feasibility

To train the model for the proposed system, A highly configured GPU is mandatory. Such types of GPU are very expensive. Hence, an alternative available to this is 'Google collab' which gives a free GPU as well as TPU as a cloud platform.

4.3. Project Resources

4.3.1. Hardware Requirements

Training machine learning model is a graphic intensive task. Images being in the form of matrix require a graphic card, hence the minimum system requirement is as follows-

- Quad core Intel Core i5 or higher
- Nvidia GeForce GTX 1050 or higher
- 8GB of RAM or higher
- 256GB of storage

4.3.2. Software Requirements

Software requirements for the project include various libraries in python which can be imported in any operating system – Linux, Windows or Mac.

Different libraries required are-

- Tensorflow
- Keras
- Pillow
- Numpy

Apart from the libraries, we also need a cloud-based platform which is capable of running and testing machine learning model. For this we use Google Colab which is a free cloud-based platform which provides a free GPU and a TPU as well if needed.

4.3.3. Operational Requirements

To operate this system, the user only requires a computer system with an active internet connection.

4.4. Project Estimation

4.4.1. COCOMO Estimation Model

The consideration of Basic COCOMO Model to estimate the development time of the project is taken. Evaluating the parameters like size of code, team size, developer experience, working environment, innovation and deadline it can be said that we are following the organic development mode as per COCOMO Model.

The estimation of the project to be of 25 KLOC (Kilo Lines of Code)

The basic COCOMO equations take the form:

- Effort applied = a* [man-months]
- Development time = c * (effort applied)b [months]
- People Required = Effort applied÷ Development time [count]

Therefore, for our project we have,

Effort = 2.4 * (25)1.05 ~ 70 person - months

Development Time = 2.5 * (70)0.38 ~ 12 months

Thus, from the COCOMO Model it may be estimated that the project can be completed in 12 months as per our project management schedule.

4.4.2. Functional Point Analysis

Table 4.1. Functional Point Analysis 1

No.	Parameter	Count	Simple	Average	Complex	Count Total
1.	No of user inputs	20	3	4	6	80
2.	No of user outputs	30	4	5	7	150
3.	No of inquiries	18	3	4	6	72
4.	No of files accessed or modified	40	7	10	15	400
5.	No of external interface referenced	5	5	7	10	35

Total = 737

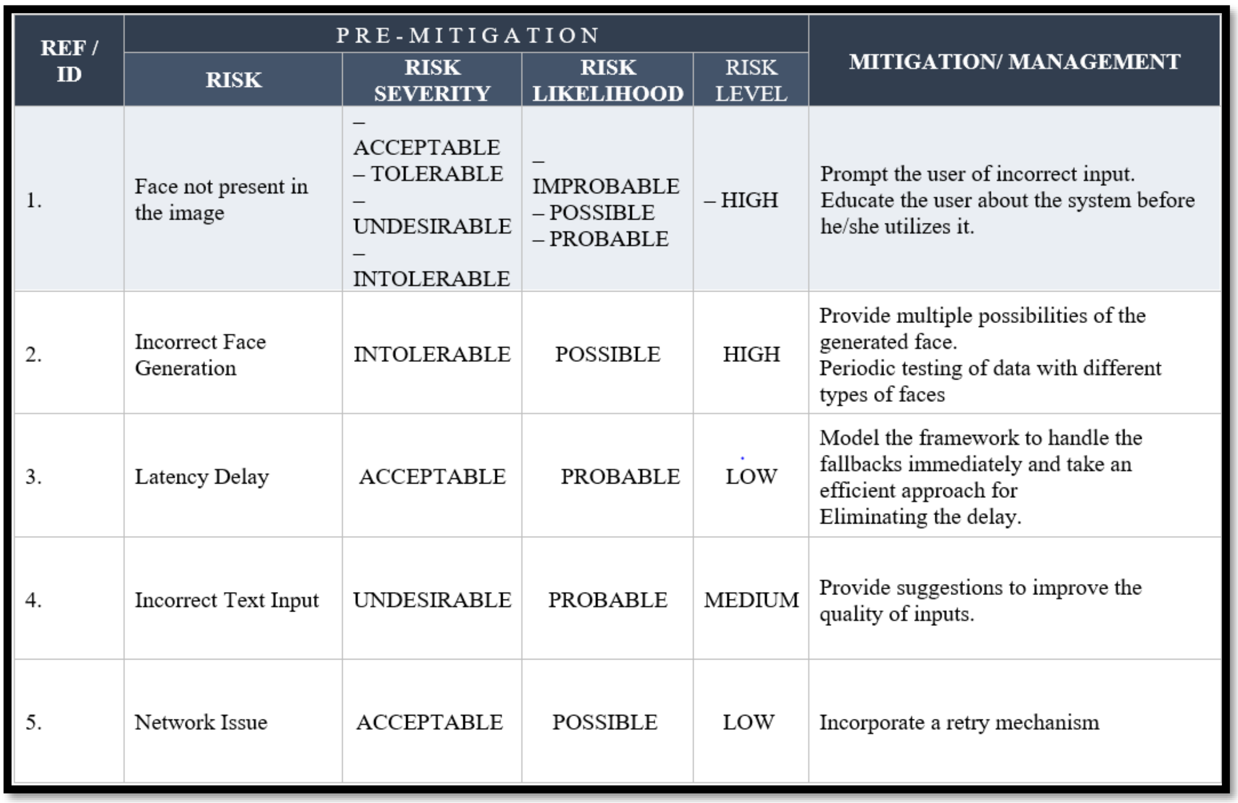
Table 4.2. Functional Point analysis 2

Number	Complexity Weighting Factor	Value
1	Backup and recovery	3
2	Data Communication	5
3	Distributed Processing	1
4	Performance Critical	4
5	Existing operating environment	5
6	On-line data entry	3
7	Input Transaction over multiple screens	3
8	Master files updated online	2
9	Information domain values complex	5
10	Internal Processing Complex	5
11	Code designed for reuse	5
12	Conversion / installation in design	4
13	Multiple installations	5
14	Application designed for change	5
	Total complexity adjustment value	55

Function Point (FP) = Total Count * [0.65 + 0.01 * ∑Fi]

= 737 * [0.65 + 0.01*55]
= **884.4**

4.5. Risk Management Mitigation Planning



REF / ID	PRE-MITIGATION				MITIGATION/ MANAGEMENT
	RISK	RISK SEVERITY	RISK LIKELIHOOD	RISK LEVEL	
1.	Face not present in the image	– ACCEPTABLE – TOLERABLE – UNDESIRABLE – INTOLERABLE	– IMPROBABLE – POSSIBLE – PROBABLE	– HIGH	Prompt the user of incorrect input. Educate the user about the system before he/she utilizes it.
2.	Incorrect Face Generation	INTOLERABLE	POSSIBLE	HIGH	Provide multiple possibilities of the generated face. Periodic testing of data with different types of faces
3.	Latency Delay	ACCEPTABLE	PROBABLE	LOW	Model the framework to handle the fallbacks immediately and take an efficient approach for Eliminating the delay.
4.	Incorrect Text Input	UNDESIRABLE	PROBABLE	MEDIUM	Provide suggestions to improve the quality of inputs.
5.	Network Issue	ACCEPTABLE	POSSIBLE	LOW	Incorporate a retry mechanism

Figure 4.3. Risk Management Mitigation Planning

Therefore, we saw the proceedings of the project for the duration of 1.5 years and also successfully made project estimations.

Chapter 5

System Design

In this chapter, we have produced various UML diagrams such as Use case and activity diagram. They provide a layout and a flow to the user for navigating and also give more pictorial information about the software. This chapter also includes system architecture which gives a detailed view of all the modules and layers of our project.

5.1. Design Diagram

5.1.1. UML Diagrams

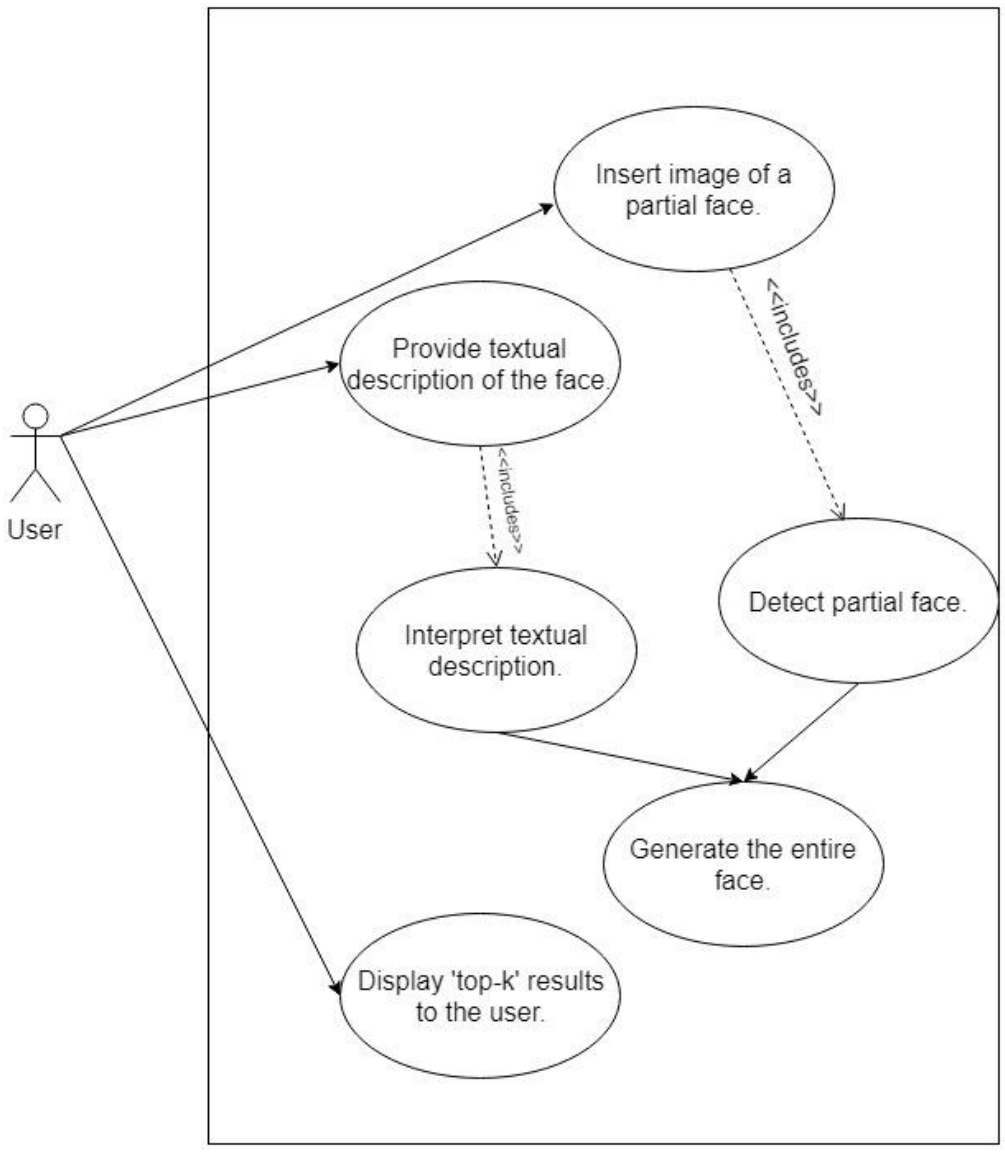

Insert image of a partial face.
Provide textual description of the face.
User
<<includes>>
<<includes>>
Interpret textual description.
Detect partial face.
Generate the entire face.
Display 'top-k' results to the user.

Figure 5.1. Use Case Diagram

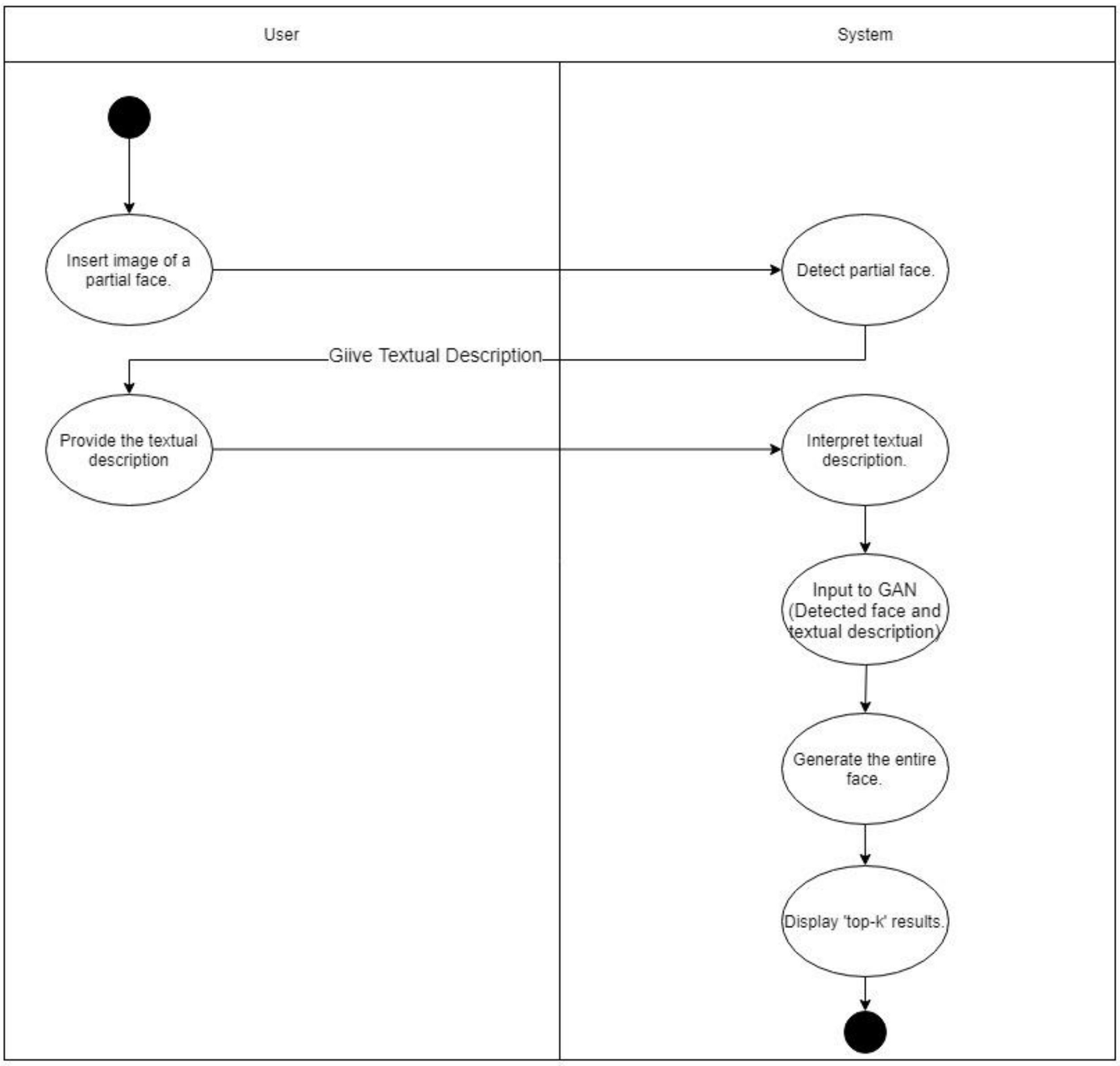

User
System
Insert image of a partial face.
Detect partial face.
Giive Textual Description
Provide the textual description
Interpret textual description.
Input to GAN (Detected face and textual description)
Generate the entire face.
Display 'top-k' results.

Figure 5.2. Activity Diagram

5.2. System Architecture

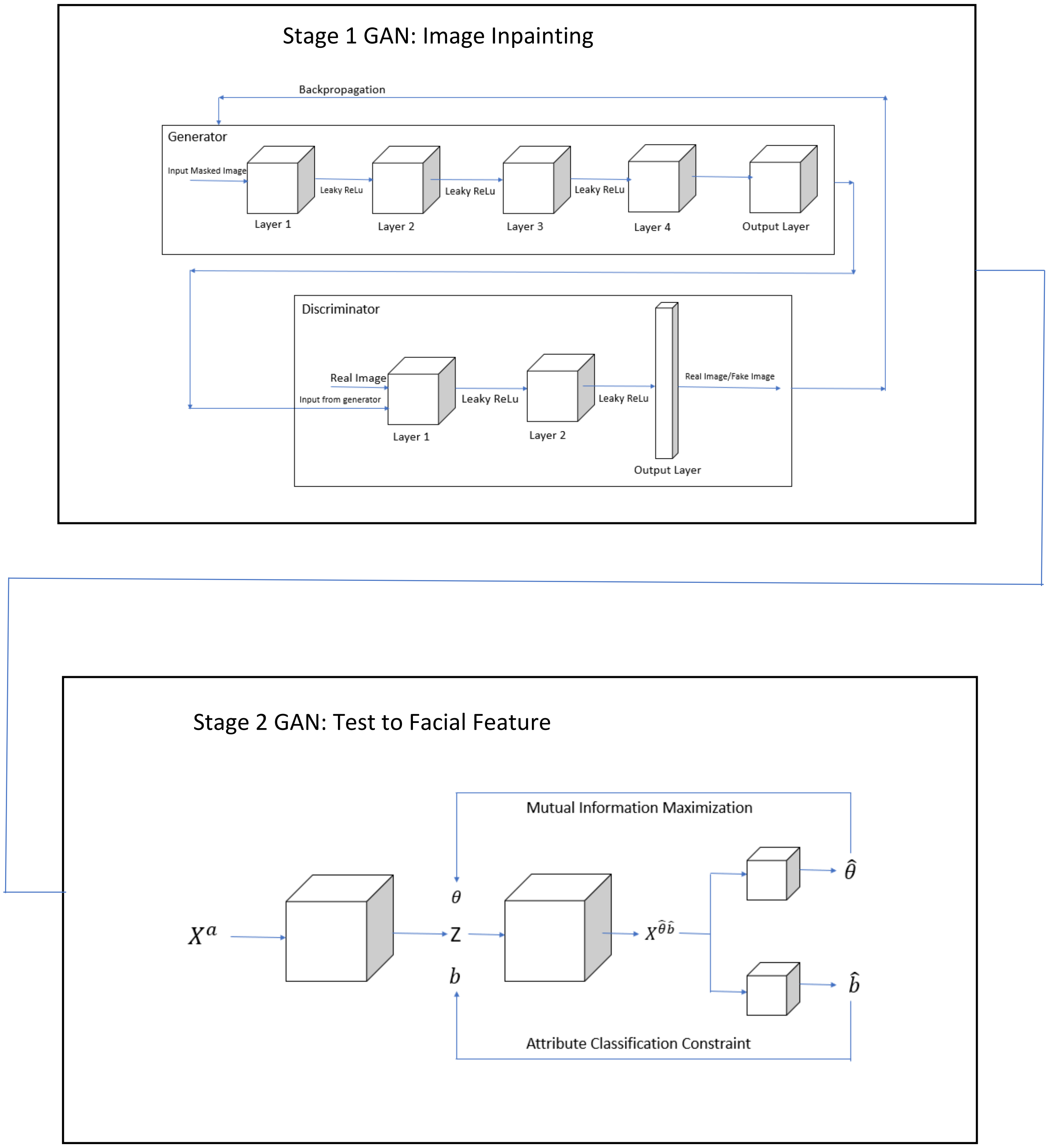


Figure 5.3. System Architecture

Chapter 6

Implementation

In this chapter we have discussed the working of our system. As described earlier in our system architecture there are two stages of GAN with a face detection module which are discussed in detail below. We have also listed in detail the tools used for this project.

6.1. Working of System

Face Detection: Face detection in our project is implemented using YOLO algorithm. It detects faces present in the input images. Unlike most common applications of YOLO which is object detection in Self Driving Vehicles, we have trained this model to detect faces in an input image. YOLO is a latest algorithm which uses convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. This algorithm has provided exceptional results in detecting faces when a part of the face was masked.

Stage – 1 Generator: The Generator Network of Stage 1 begins with a dense layer that takes the masked image as input. This layer applies Leaky ReLU activation on the input image. The desired output of the generator is an image of dimension 28x28x3. To achieve this output, 3 convolutional layers with 128, 64 and 3 filters respectively of dimension 5x5 each and using same padding are applied. This generates the desired output image of dimension 28x28x3. After each convolutional layer, we use a leaky ReLU activation to activate the inputs received from the previous layer. This generated image is passed on the Discriminator of first stage.

Stage 1 input Stage 1 output of Generator

→

Stage – 1 Discriminator: The Discriminator of first unit begins with a Convolutional Layer with 64 filters of size 5x5 and same padding. Leaky ReLU is again used for activation. This layer is followed by another conv-relu pair with 128 filters of dimension 5x5. This is then connected to a dense layer consisting of only one output unit to classify whether generated image is real or fake.

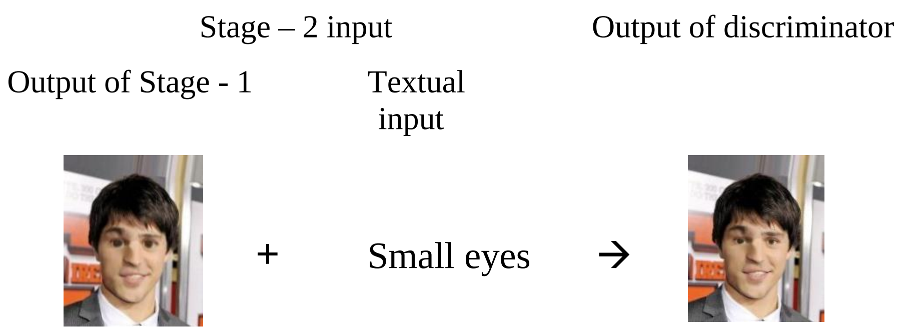
Stage – 1 Discriminator input Output of discriminator

Output of Generator Original Image

+

→ Real or Fake (1/0)

Stage – 2: This module comprises of two basic subnetworks i.e. Encoder (Genc) – Decoder (Gdec) along with Classifier (C) – Discriminator (D). This module will map the textual description of the facial features to the generated image. Stage 2 GAN removes the strict attribute independent constraint from the latent representation and just applies the attribute classification constraint to the generated image to assure the correct change of attribute. The encoder Genc which takes an image as an input and outputs a vector is a stack of convolution layers. Out of the two de-coders, one decoder Gdec converts the vector and original attributes back into image. Reconstruction loss is calculated at this stage which helps us understand the generation quality of the generator network. The other decoder Gdec converts the vector and desired attributes to the required image. The decoder Gdec is a stack of transposed convolutional layers. The required image is then passed through a discriminator and a classifier. Discriminator is followed by a fully connected layer which calculates the adversarial loss needed to improve the model. Classifier on the other hand classifies the generated image as real or fake. Along with addressing the peculiarities of the face, this approach will also handle minor errors in semantic representation of stage 1 GAN.



6.2. Algorithm

There are two different algorithms used in system. Explanation of each algorithm based on the modules are as follows:

Image pre-processing Unit: This unit consists of Face detection implemented using YOLO algorithm to detect faces present in the input images. Unlike most common applications of YOLO which is object detection in Self Driving Vehicles, we have trained this model to detect faces in an input image. YOLO is a latest algorithm which uses convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. This algorithm has provided exceptional results in detecting faces when a part of the face was masked.

Generative Adversarial Networks (GAN): GAN is the main algorithm used for the project. A generative adversarial network (GAN) is a class of machine learning systems which consists of two neural networks which contest with each other in a game. Given a training set, this technique learns to generate new data with the same statistics as the training set. For example, a GAN trained on photographs can generate new photographs that look at least superficially authentic to human observers, having many realistic characteristics.

We are using two stages of GAN to achieve desired results. First stage of GAN uses Convolutional Neural Network as a generator network to generate image from its masked representation and the second unit of GAN uses an encoder based on GAN to understand textual description and apply it on the image generated by the first unit.

6.3. Tools and Technology

Python: Python is an interpreted, high-level, general-purpose programming language. Python is commonly used in artificial intelligence projects with the help of libraries like TensorFlow, Keras, Pytorch and Scikit-learn. As a scripting language with modular architecture, simple syntax and rich text processing tools, Python is often used for natural language processing.

NumPy: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

Pandas: Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

Pillow: Pillow is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux.

TensorFlow: TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

Keras: Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

Chapter 7

Testing and Results

In this chapter we have discussed various conditions to test the credibility of our system and also provided results of the tests conducted. Thus, we have inferenced whether our system performs to expectations or no. If no, details regarding improvement for the same are provided. Also, the results we achieved are given in this chapter.

7.1. Test Plan

7.1.1. Purpose of the Test Plan Document

The objective of this test plan document is to specify various scenario in which the accuracy of the developed system will be tested. This document also specifies the results that render the test successful and declare the system as working in normal state, along with the results that render the test unsuccessful and demand for improvement in the system. Further, we specify various test cases that will be used to determine the ability of the system and providing testing deliverables which serve as the feedback of the testing that includes necessary steps that needs to be carried out in the system to make it efficient.

7.1.2. Items to be Tested

List and Describe the items/features/functions to be tested that are within the scope of this test plan. Include a description of how they will be tested, when, by whom, and to what quality standards. Also include a description of those items agreed not to be tested.

Table 7.1. Items to be Tested

Item to Test	Test Description	Test Date	Responsibility
YOLO face detection	**This test will check whether our system is able to detect a person's face in an image given as input**		**To identify various scenarios in which the system gives positive results and the scenarios in which the system gives negative results.**
Face Generation	**This test will check the ability of the GAN module to successfully generate a person's face given a partial face as input.**		**To identify various scenarios in which the system recreates a person's face with 75% accuracy and the scenarios in which the system fails to generate a person's face.**

7.1.3. Test Approach(s)

Table 7.2. Testing Approaches

Module(s)	**Test Approach**	**Remarks**
YOLO Face Detection	**P**rovide faces of a person with different orientation, different visible features of face and with different sizes of masks on face.	**This test can successfully determine various angles of face and visible pixels of an image that our system can detect.**
Face Completion Module	**Provide masked face of people with changing size of masks.**	**This test with determine whether the desired scope of face completion module is being achieved or no.**

7.1.4. Test Regulatory/ Mandate Criteria

The system must be tested against face of a person which is half covered. This test will verify whether our system is able to generate faces in situation where only one half of the face is available. This will verify 50% scope of the project.

7.1.5. Pass/Fail Criteria

Test is considered as successfully passed when the system is able to detect partial face in a picture and is able to generate the hidden part of the face with 75% accuracy or more.

There are two failing criteria: 1) If the test is not able to detect partial face in the input image and 2) It is unable to generate a face given its hidden self.

7.1.6. Test Entry/ Exit Criteria

Entry criterion is an image of a person with his partial face hidden and exit criterion is the result of the test whose conditions are described in the session above.

7.2. Testing Strategy

7.2.1. Unit Testing

Table 7.3. Unit Testing of Face Detection Module

Test Scenario ID:		T101		Test Case ID:	TC101	
Test Priority:		Moderate				
Test Case Description:		Testing of Face Detection Module				
Test Pre-Requisite:		Image with face of a person that is partial hidden		Test Post-Requisite:		
Sr. No	Action / Module	Input	Expected Input	Actual Output	Test Result (Pass / Fail)	Test Comments
1	Detect persons face	Image with persons face partially hidden	Image with persons face partially hidden	Image with bounding boxes around detected persons face	Pass	Verification of proper working of Face detection Module

Table 7.4. Unit Testing of Face Completion Module

Test Scenario ID:		T102		Test Case ID:	TC102	
Test Priority:		Highest				
Test Case Description:		Testing of Face Completion Module				
Test Pre-Requisite:		Image with face of a person that is partial hidden		Test Post-Requisite:		
Sr. No	Action / Module	Input	Expected Input	Actual Output	Test Result (Pass / Fail)	Test Comments
1	Detect persons face	Image with persons face partially hidden	Image with persons face partially hidden	Image with the hidden region of a person’s face filled up to 75% accuracy	Pass	Verification of proper working of Face completion Module

7.2.2. Integration Testing

Table 7.5. Integration Testing

Test case ID	Test Case Objective	Approach used	*Exit Criteria	Remark
1	Proper working of all the modules concurrently	Give Input image to first module, take the output of first module and give it to 2nd module. Check the output image of the second module.	Image with filled missing region with or without 75% accuracy	Verifies proper working of all the modules concurrently

7.2.3. System Testing

Table 7.6. System Testing

Test case ID	System Testing Parameters / Feature	Action Taken	*Exit Criteria	Remark
1	Full system Test	Improvement of the results achieved by testing	Positive or negative results from the system	Helps to verify the credibility of the system.

7.3. Test Planning

7.3.1. Test Schedule

Table 7.7. Test Planning Schedule

*Task Name	Start	Finish	Effort	Comments
Test Planning	02/02/2020	05/02/2020	2 days	Created a plan for testing
Review Requirements documents	02/02/2020	04/02/2020	2 days	Reviewed Testing Requirements
Create initial test estimates	04/02/2020	05/02/2020	1 days	Created Initial testing criterion
Staff and train new test resources	06/02/2020	10/02/2020	4 days	Gathered testing resources
First deploy to QA test environment	11/02/2020	12/02/2020	2 days	Deployed Testing environment
Functional testing – Iteration 1	13/02/2020	16/02/2020	3 days	Carried out Functional Testing 1
Iteration 2 deploy to QA test environment	17/02/2020	20/02/2020	3 days	Collected results
Functional testing – Iteration 2	21/02/2020	26/02/2020	5 days	Carried out Functional Testing 2
System testing	27/02/2020	28/02/2020	2 days	Carried Out System Testing
Resolution of final defects and final build testing	01/03/2020	02/03/2020	2 days	Gathered Feedbacks from various tests
Deploy to Staging environment	03/03/2020	05/03/2020	3 days	Finalized product

7.3.2. Test Deliverables

The testing phase helped us identify different conditions in which the system was able to perform as expected and situations where the system gave negative results. We identified that the system was no able to detect faces when more than half of the face was covered and when the faces were not aligned. Also, the face generation module failed when the faces were not center cropped. To avoid these errors, we have to use image augmentation technique to provide different variants of the same image. Also using a bigger dataset can help us considerably.

7.3. Experimental Results

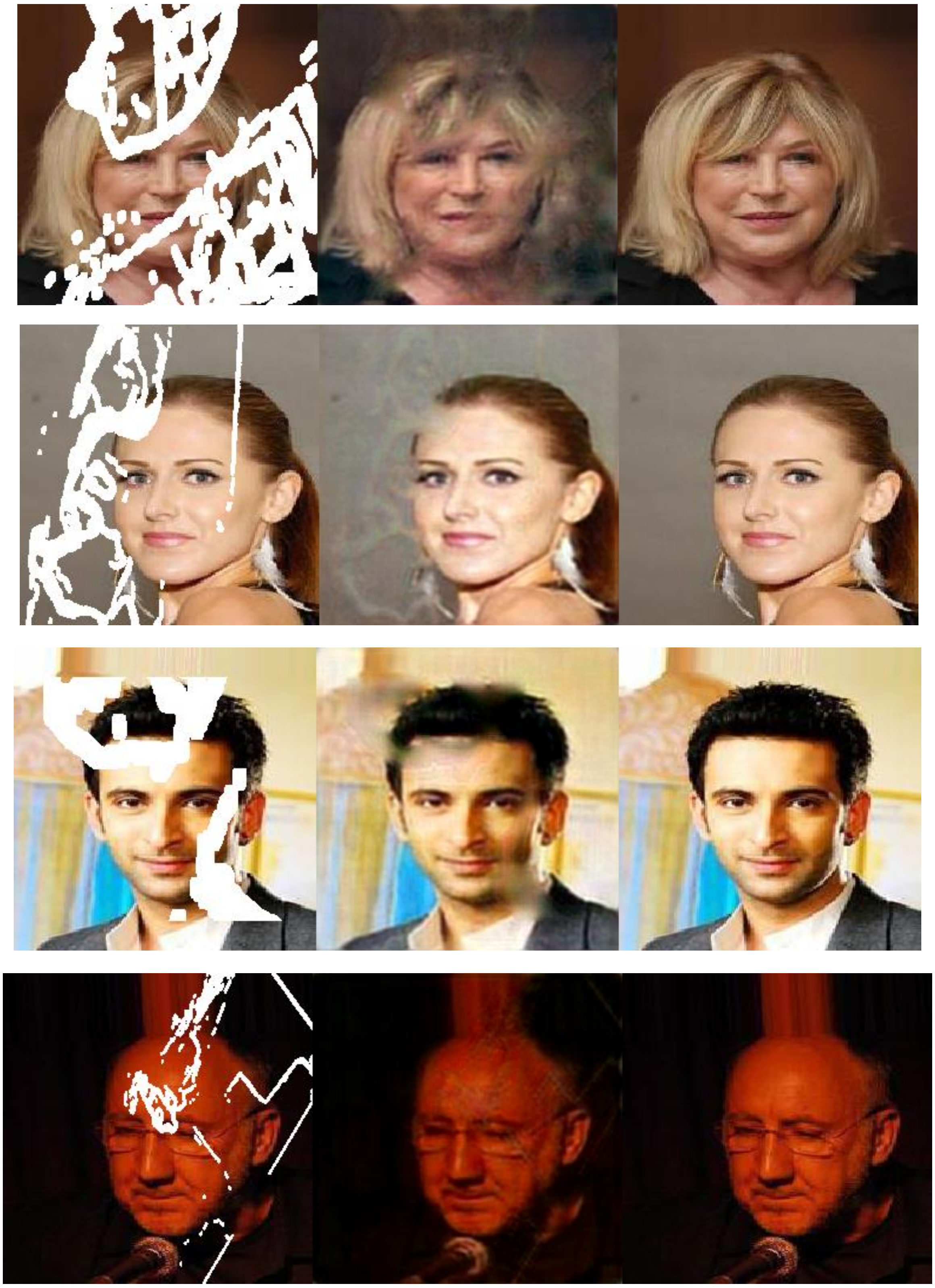


Figure 7.1. Results

Chapter 8

Conclusion

In this project we proposed a concept of using two Generative Adversarial Networks in order to implement face inpainting. The proposed model can successfully synthesize semantically valid and visually plausible contents for the missing facial key parts from random noise. Also, with the help of textual description, it was possible to in-crease the accuracy of the image that is in-painted. Our model is found to give better performance than many of the existing models related to this domain. Hence, we hope this model will prove to be of significant help to crime investigation department.

Chapter 9

Future Scope

Although our model is able to generate semantically plausible and visually pleasing contents, it has some limitations. The faces in the CelebA dataset are roughly cropped and aligned. We implement various data augmentation to improve the robustness of learning, but find our model still cannot handle some unaligned faces well. The unpleasant synthesized contents indicate that the network does not recognize the position/orientation of the face and its corresponding components. This issue can be alleviated with 3D data augmentation.

References:

1. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. In NIPS, 2014.

2. Yijun Li , Sifei Liu , Jimei Yang , and Ming-Hsuan Yang. Generative Face Completion. In arXiv preprint arXiv:1704.05838, 2017.

3. Kushagr Gupta, Suleman Kazi, Terry Kong. DeepPaint: A Tool for Image Inpainting. At Standford, 2016.

4. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee. Generative Adversarial Text to Image Synthesis. In arXiv preprintvarXiv:1605.05396v2, 2016.

5. Xinchen Yan1, Jimei Yang2, Kihyuk Sohn, Honglak Lee. Attribute2Image: Conditional Image Generation from Visual Attributes. In arXiv preprint arXiv:1512.00570v2, 2016.

6. Han Zhang , Tao Xu , Hongsheng Li , Shaoting Zhang , Xiaogang Wang , Xiaolei Huang , Dimitris Metaxas. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. In IEEE International Conference on Computer Vision, 2017.

7. Xu Ouyang , Xi Zhang , Di Ma, Gady Agam. Generating Image Sequence from Description with LSTM Conditional GAN. In 2018 24th International Conference on Pattern Recognition, 2018.

8. Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan and Xilin Chen. AttGAN: Facial Attribute Editing by Only Changing What You Want. In IEEE Transactions on Image Processing, 2019.

9. Bernd Heisele, Thomas Serre, T.Poggio. A Component-based Framework for Face Detection and Identification. In Springer International Journal of Computer Vision, 2006.

10. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In IEEE Conference on Computer Vision and Pattern Recognition, 2016.

11. Y. Wexler, E. Shechtman and M. Irani. Space-Time Video Completion. In journal preprint from TPAMI, 2007.

12. Omar Elharroussa, Noor Almaadeeda, Somaya Al-Maadeeda, Younes Akbaria. Image inpainting: A review. In arXiv preprint arXiv:1909.06399, 2019.

13. Piotr Teterwak, Aaron Sarna, Dilip Krishnan, Aaron Maschinot, David Belanger, Ce Liu, William T. Freeman. Boundless: Generative Adversarial Networks for Image Extension. In arXiv preprint arXiv:1908.07007v1, 2019.

14. Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In IEEE International Conference on Computer Vision (ICCV), 2015.

15. M. Mirza and S. Osindero. Conditional generative adversarial nets. In arXiv preprint arXiv:1411.1784, 2014.

16. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al. Tensorflow: A system for large-scale machine learning.

17. D. Kingma and J. Ba. Adam: A method for stochastic optimization. In International Conference on Learning Representations (ICLR), 2015.

18. S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning (ICML), 2015