CAPSTONE PROJECT REPORT

(Project Term July-November 2019)

Image Compression using CNNs

Submitted by

Sabid Ansari	**Registration Number:11617037**
Adarsh Khare	**Registration Number:11612923**
Vamja Chintan	**Registration Number:11610527**
Tarun Singh Mahar	**Registration Number:11608343**

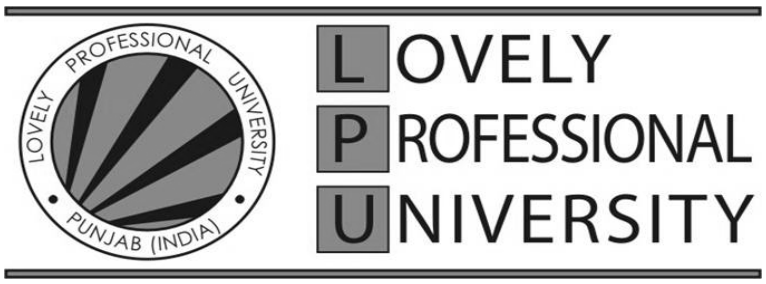
Project Group Number: CSERGC0321

Course Code: CSE-439

Under the Guidance of

Samreen Fayaz

School of Computer Science and Engineering



PAC Form

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering (SCSE)

Program : P132::B.Tech. (Computer Science & Engineering)

COURSE CODE : CSE439 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CSERGC0321

Supervisor Name : Samreen Fayaz **UID :** 25435 **Designation :** Assistant Professor

Qualification : _____ **Research Experience :** _____

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Sabid Ansari	11617037	2016	K1630	9872448321
2	Adarsh Khare	11612923	2016	K1619	9115513906
3	Vamja Chintan	11610527	2016	K1631	9515782348
4	Tarun Singh Mahar	11608343	2016	K1621	9760856822

SPECIALIZATION AREA : Software Engineering **Supervisor Signature:** _____

PROPOSED TOPIC : Data Compression using Machine Learning

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	**Parameter**	**Rating (out of 10)**
1	Project Novelty: Potential of the project to create new knowledge	6.85
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	7.00
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.15
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.85
5	Social Applicability: Project work intends to solve a practical problem.	6.62
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.15

PAC Committee Members		
PAC Member (HOD/Chairperson) Name: Dalwinder Singh	UID: 11265	Recommended (Y/N): Yes
PAC Member (Allied) Name: Asha Rani	UID: 11332	Recommended (Y/N): Yes
PAC Member 3 Name: Parampreet Kaur	UID: 18758	Recommended (Y/N): Yes

Final Topic Approved by PAC: **Data Compression using Machine Learning**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 11024::Amandeep Nagpal **Approval Date:** 04 May 2019

11/14/2019 12:36:24 PM

DECLARATION

We hereby declare that the project work entitled "**Image Compression using CNN**" is an authentic record of our own work carried out as requirements of Capstone Project for the award of B.Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara, under the guidance of Samreen Fayaz during July to November 2019. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Project Group Number: KC321

Name of Student 1: Sabid Ansari
Registration Number: 11617037

Name of Student 2: Adarsh Khare
Registration Number: 11612923

Name of Student 3: Vamja Chintan
Registration Number: 11610527

Name of Student 4: Tarun Singh Mahar
Registration Number: 11608343

(Signature of Student 1)

Date:

(Signature of Student 2)

Date:

(Signature of Student 3)

Date:

(Signature of Student 4)

Date:

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Capstone Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other degree at any University. The Capstone Project is fit for the submission and partial fulfillment of the conditions for the award of B.Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara.

Samreen Fayaz

Assistant Professor
School of Computer Science and Engineering,
Lovely Professional University,
Phagwara, Punjab.

Date:

ACKNOWLEDGEMENT

We humbly take this opportunity to present our votes of thanks to all those guideposts who really acted as lightening pillars to enlighten our way throughout this project that has led to successful and satisfactory completion of this study.

We are grateful to our mentor Samreen Fayaz for providing us with an opportunity to undertake this project in this university and providing us with all the bright and innovative ideas for making our project a really worthwhile of running in an organization. We are highly thankful for his active support, valuable time and advice, whole-hearted guidance, sincere cooperation and pains-taking involvement during the study and in completing the capstone project within the time stipulated.

We are thankful to all those, particularly our friends , who have been instrumental in creating proper, healthy and conductive environment and including new and fresh innovative ideas for us during the project, without their help, it would have been extremely difficult for us to prepare the project in a time bound framework.

TABLE OF CONTENTS

1. INTRODUCTION

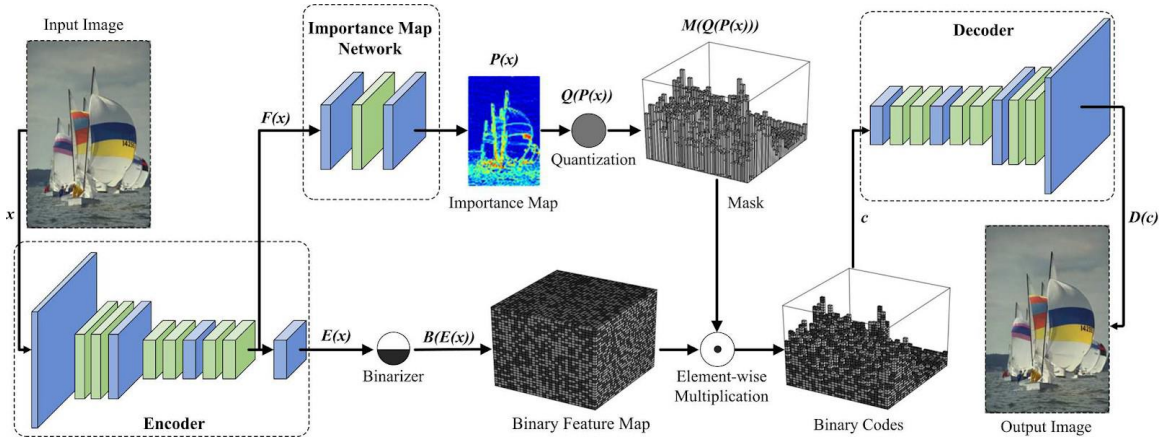
In the world today, more and more images are being captured, with the advent of Internet Of Thing (IOT) and automation, even greater number of synthetic images are being generated by the machines. The tools not only facilitate image capture, but also contribute to higher resolution images. These images occupy more space and require greater bandwidth for transfer as compared to a low resolution image. Storage and transfer of such images is a biggest challenge we can see. In this project, we aim to find improved way to compress an image using neural networks.

Image Compression is another form of data compression which is applied to the digital images to reduce its size and the storage sufficiency. It helps to compress the size of the image which is very helpful for the maintenance and storage of an image. Many images which we see on the internet are to be uploaded with the help of image compression. It helps the person to upload the picture quickly on the web and use less space because it reduces the size of images and reduces the pixels also. The image compression doesn't compress the physical size of the images it just compress the data so that it is easily upload to the web and to be stored. In simple words, image compression is all about reducing the size of images and different graphics in bytes without degrading the much quality. After the image compression process, the image quality must be at an acceptable level. With the minimization of the size if the image, a number of images can be stored in a specific memory space. Along with this, it also helps to minimize the time required for graphics to upload on the internet or download from different websites and web pages.

1.1. Flow of Image Compression

Image Compression is performed using image compression coding. It is all about storing the bit-stream of an image into a compact file and displaying the decoded image at the monitor exactly as it was displayed with the orignal size. The figure shows complete description of the image compression process. At first, an orignal image is provided to the encoder, which is converted into a bitstream. On the encoder side, encoder network consists of three convolutions layers and three residual blocks and each residual block has two convolution layers. Then it comes to binarizer, the binarizer in the system is used for the quantization of the image data. On the other side, there is Importance Map which will be used to determine how many bits need to be allocated to a particular region. Both the parts will be connected to decorder, the network

architecture of the decoder is symmetric to that of the encoder, which is decoded in the form of an image. As the total data quantity at the decoder side is less than the encoder side, the image can be called compressed image.

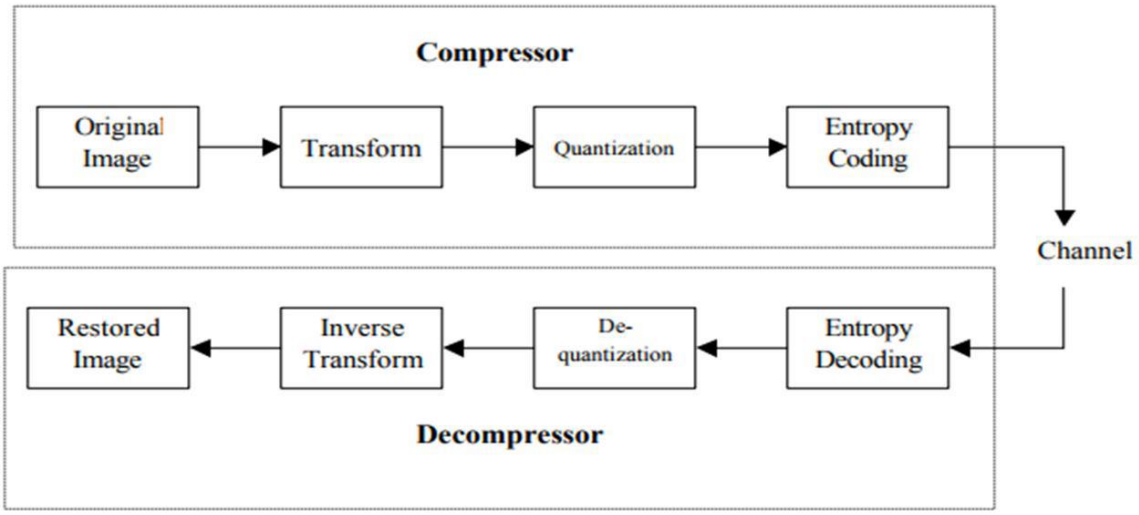


1.2. Techniques of Image Compression

There are various image compression methods have been developed over the past two decades. All these compression methods can be broadly classified into two categories including lossy or lossless image compression. A high compression ratio of approximately 50:1 can be accomplished using the lossy compression because a certain level of degradation in the image quality is allowed in the lossy image compression techniques. But it is not able to recover the original data completely. Second techniques are the lossless compression in which possibly there is no degradation in the image quality. It can completely recover the original data of the image. If we talk about the compression ratio, it is only 2:1. The complete description of both type of image compression method are described below

1.2.1. Lossy Image Compression

In the first stage of the lossy image compression technique, the interpixel redundancy is eliminated to ensure efficient packing of the information. After that, in the quantization stage, psycho-visual redundancy is removed to represent the information as few bits as possible. At last, more compression is accomplished by encoding the image from the coding redundancy.

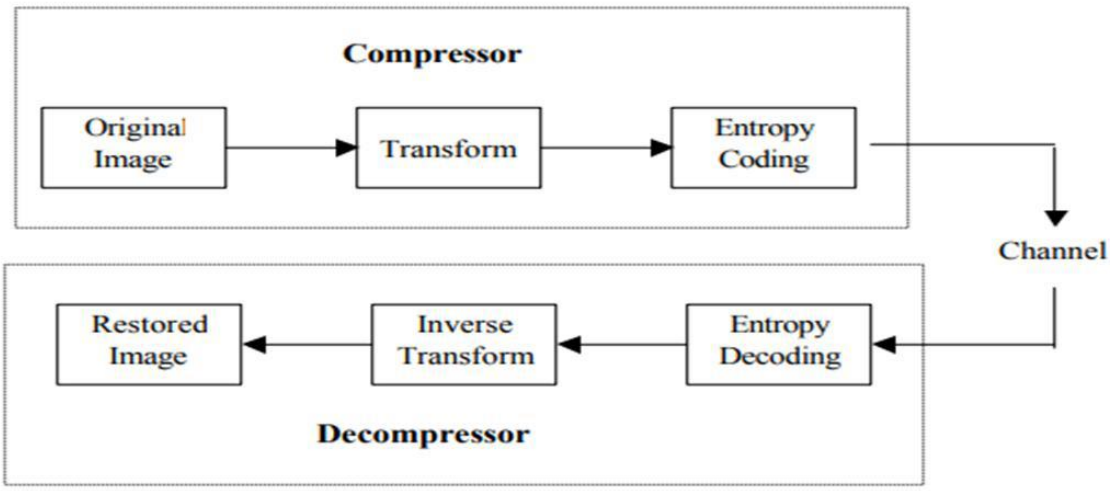


It is divided into various types:

1. Transformation Coding: It is used to change the pixel of the current or present images so that it can be stored easily and anywhere.
2. Vector Quantization: It is used to develop a fixed vector in the image to reduce the pixels present in the image.
3. Fractal Coding: It is used to decompose the image into different segments by using the technique of standard image processing, color separation and edge detection.
4. Block Truncation Coding: It is divided into blocks and many other segments to reconstruct the values in the image.

1.2.2. Lossless Image Compression

It is a two-step algorithm. In the first step, the original image is transformed to any other format to reduce the interpixel redundancy. In the second step, coding redundancy is removed using an entropy encoder. At the decoder side, a lossless decompress perform the inverse process and retrieve the original information without degrading the quality.



Here, there is no loss of information. If the necessary image is decomposed, then it can be converted to original image easily.

2. Profile of problem. Rationale/Scope of study.

The project is built on the basics of Machine Learning and the libraries of python such as TensorFlow and the concept of Convolutional Neural network, and the concept of building the Graphical user interface using python-GUI Tkinter which help us in creation of simple and easy GUI.

Based on our model which we created using 256 Caltech Image dataset using which we compress our image and keep its quality to maximum state also.

In this project we used the concept of Convolutional Neural Network which help us in compression of image and giving us the expected output.

3. Existing Project and Software's

3.1. Introduction

The concept of image compression has been part of our day to day life, but we don't know like in WhatsApp we daily send image, but it compresses the image and send it to the other user.

There are many existing software and website which help us in compressing our image, but they usually cost us our image quality. Some of the websites are

https://compressjpeg.com/,http://www.countingcharacters.com/imagecompressor-online and many more.

3.2. What's new in the system to be developed

we have introduced here machine leaning (CNN) here to use which help us in compression of our image in much more efficient manner. Simply put our model detects the object in the provided image and keeps its quality while reducing the image quality of other parts It work like human brain which focus on main part of image.

The reason for building such software, as the number is increasing on daily basis and the storage required to keep those images is also increasing so we need a compression where we can reduce our image size while keeping its quality to optimum level.

4. Problem Analysis

4.1. Product Definition:

Our product would be an image compression gui from where we must select an image file either in png format or in a jpg format after selecting a file we have to select an output folder from where the compressed image will be saved after being compressed. After the image is compressed user get the message that the image is compressed and saved in his desired folder. If anytime user wants to exit from the gui then he or she can exit the gui by clicking the exit button that is present in the gui.

Features of product:

1. It gives the compressed image with the minimum size without reducing the quality of the image file.
2. Image compression gui is very easy to use.

4.2. Feasibility Study

4.2.1. Technical Feasibility:

In Technical feasibility, we just need to see that our project is possible technically or not.

- Firstly, we needed to see that there is dataset available for our project, which is possible as we are using 256 Caltech Image Dataset.
- After the dataset we needed to see that whether the image compression is possible using CNNs or not.
- If there is no problem in training, testing and creation of our Machine Learning model than we can see that if it is possible than we must if it is possible to link our model with the GUI.

4.2.2. Economical Feasibility:

- In economic feasibility we need to see whether our project will cost us money or not and whether our project will give us some profit in return also.
- Image compression is very economical as the number of images are also increasing at very high rate so we will need storage to store them.
- So, by compressing size of image with keeping quality of image to its optimum level we can save our storage/memory and save lot of storage which will directly save our cost.

4.2.3. Operational Feasibility:

After the completion of the project we need to have some operations that need to be done time to time such as updating our dataset so that our model can learn more and more.

Train our model on more dataset which will help our model learn more as a result it will improve our image compression.

4.3. Project Plan:

- **IDEA OF THE PROJECT AND TECHNOLOGY**

TASK	Idea and Feasibility of The Project
START FROM	15/02/2019
NO. OF DAYS	28
END AT	10/03/2019

- **Requirement analysis**

TASK	Requirement analysis
START FROM	11/03/2019
NO. OF DAYS	29
END AT	10/05/2019

- **Implementation: Coding**

TASK	Coding
START FROM	11/07/2016
NO. OF DAYS	50
END AT	01/09/2016

- **Implementation: Designing**

TASK	Design
START FROM	02/09/2019
NO. OF DAYS	10

END AT	12/09/2019

- **Testing**

TASK	Testing and Fixing the Issues
START FROM	13/09/2019
NO. OF DAYS	10
END AT	23/09/2019

5. SOFTWARE REQUIREMENT ANALYSIS

5.1. Introduction:

For our projects, we would be using most of the open source libraries and frameworks. We would also be doing coding from scratch whenever required. Since it is the team projects we would be building parts of projects on our system.

5.2. GENERAL DESCRIPTION

PYTHON:

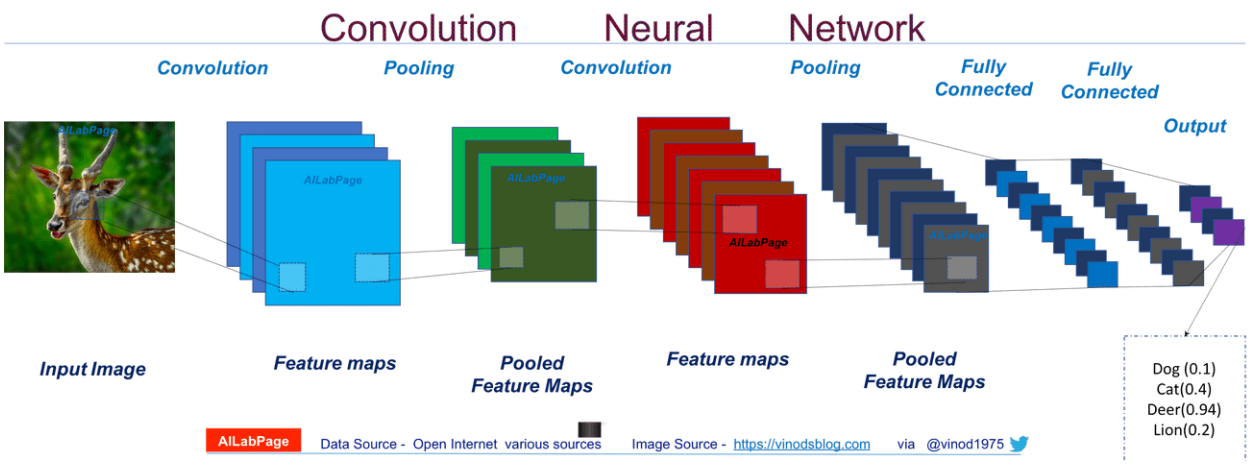
Python is an interpreted, high-level, general-purpose programming language. It is created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability through use of significant whitespace. Its object–oriented approach help programmers to write logical and clear code for small and large scale projects. Python supports multiple paradigms including object oriented, functional programming and procedural. We are using python 2.7 for this project. The main advantage of using python 2.7 over python v3.6 is that there's an awful quantity of useful libraries that haven't been ported to 3.x yet.

Convolutional neural network:

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do image recognition, image classifications. Objects detection, recognition faces etc. are some of the areas where CNNs are widely used. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply SoftMax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

Convolution Layer

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernal. Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example shows various convolution image after applying different types of filters (Kernels).



NUMPY:

NumPy is a library for the Python programming language, adding support for large, multi-dimensional metrics and array, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy ancestor Numeric was originally created by Jim

Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy. NumPy is the fundamental package for scientific computing with Python. It contains various features including these important once:

1. Tools for integrating/C++ and Fortran code
2. Sophisticated(broadcasting) functions
3. Useful Fourier transform, random number capabilities and linear algebra
4. A powerful N-dimensional array object

TENSOR FLOW:

Tensor-Flow was developed by the Google Brain team for internet Google use. It was released on November 9,2015 under the Apache License 2.0. Tensor-Flow is a free and open-source software library for differentiable and dataflow programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. Tensor flow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. TensorFlow can run on multiple CPUs and GPUs. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. Tensor flow is available on 64-bit macOS, Linux, Windows and mobile computing platforms including iOS and Android.

TensorFlow architecture works in three parts:

- Pre-processing the data
- Build the model
- Train and estimate the model

PANDAS:

Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Pandas is a

NumFOCUS sponsored project. Pandas is a software library written for the Python programming language for data manipulation and analysis.

SCIKIT-IMAGE

Scikit-image is an open-source image processing library for the python programming language. It includes algorithms for geometric, color space, filtering, feature selection, segmentation, transformations and more. It is designed to interoperate with the Python numerical and scientific libraries SciPy as NumPy. It was released in August 2009.The scikit-image project started as scikits. image, by Stefan van der Walt. Scikit-image has also been active in the Google Summer of Code.

TKINTER

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, Tkinter is commonly used method. Python with Tkinter is the easiest and fastest way to create a GUI application. Tkinter is the standard Python interface to the Tk GUI toolkit. Tkinter is included with the standard Microsoft, Mac OS, Linux and Windows installs of Python. Tkinter is free software released under the Python license.

Creating a GUI using Tkinter in an easy task:

1. Import the module -Tkinter
2. Create the main window
3. Add any no of widgets to the main window
4. Apply the event Trigger on the widgets.

There are no of widgets which you can put in your Tkinter application.

Some of the major widgets are:

1. Button: To add a button in application, this widget is used.

   The general syntax is:

   a=Button (root, option=value)

2. Canvas: It is used to draw pictures and other complex layout like graphics, text and widgets.

   The general syntax is:

   a=Canvas (root, option=value)

   root is the parameter used to represent the parent window

3. Label: It refers to the display box where you can put any text image or text which can be displayed on the main window.

   The general syntax is:

   a=Label (root, option=value)

   Option can be image, width, height, command

4. Frame: It acts as a container to hold the widgets. It is used for grouping and organizing the widgets. The general syntax is:

   a=Frame (root, option=value)

5.3. Specific Requirements:

5.3.1. Software and Hardware requirement:

5.3.1.1 Environment:

• Operating System: - Ubuntu 16.04 or Higher

• Tools: PyCharm, Python v2.7

• User Interface: Python using Tkinter

Hardware requirements:

Number	Description
1	PC with 4GB RAM and 2GB graphic card (Minimum)

Software requirements:

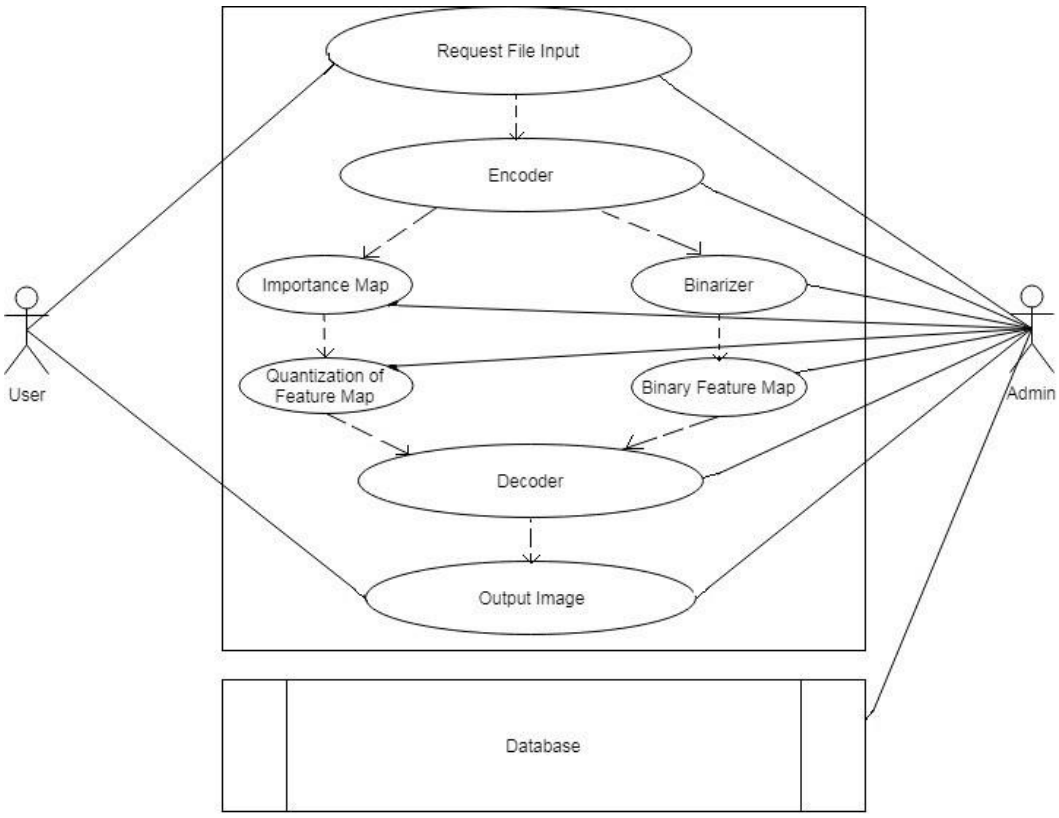
Number	Description
1	Ubuntu 16.04 or Higher
2	Python V2.7
3	PyCharm

6. Design

The design of this application is very simple and easily understandable.

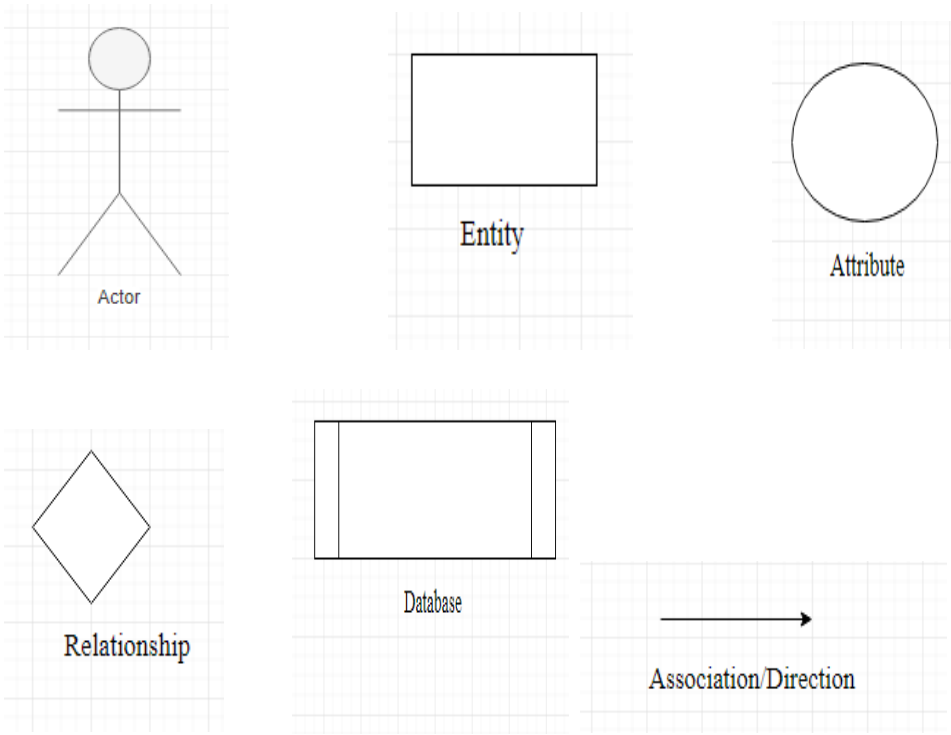
6.1. System Design

This part tells about the way the whole system is designed and it is explained using Unified Modeling Language (UML) Diagram.

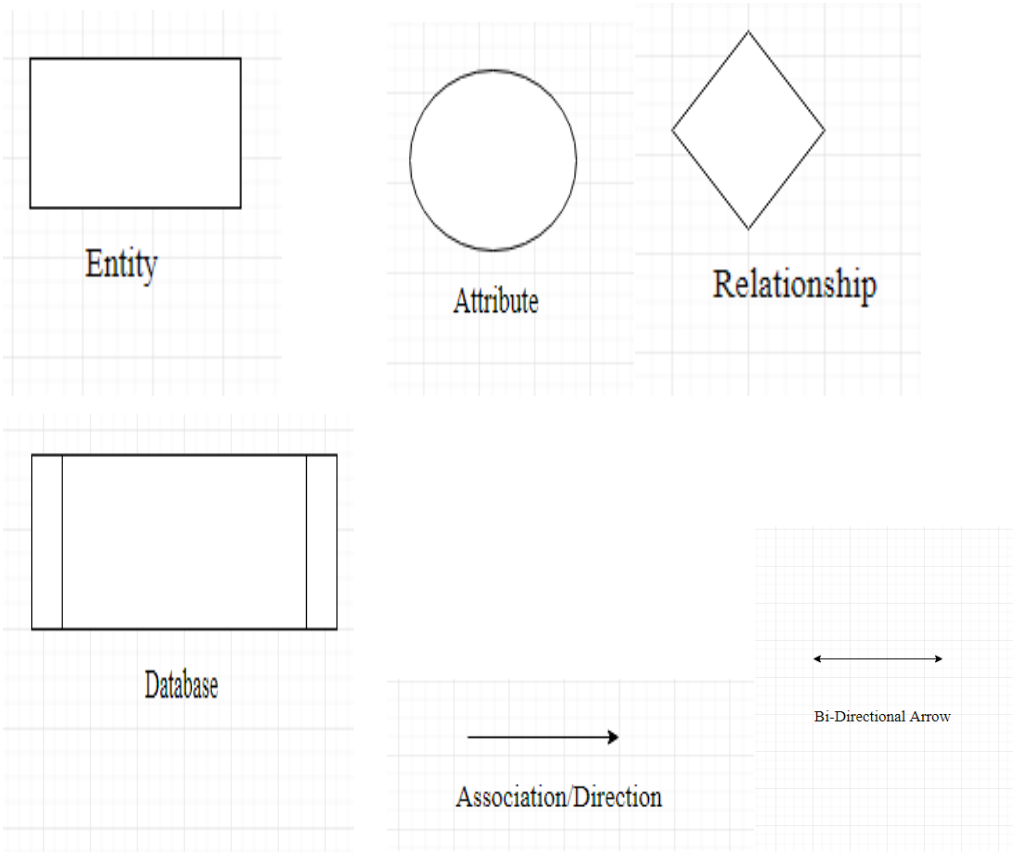


6.2. Design Notations

Design notations used in E-R diagrams are:



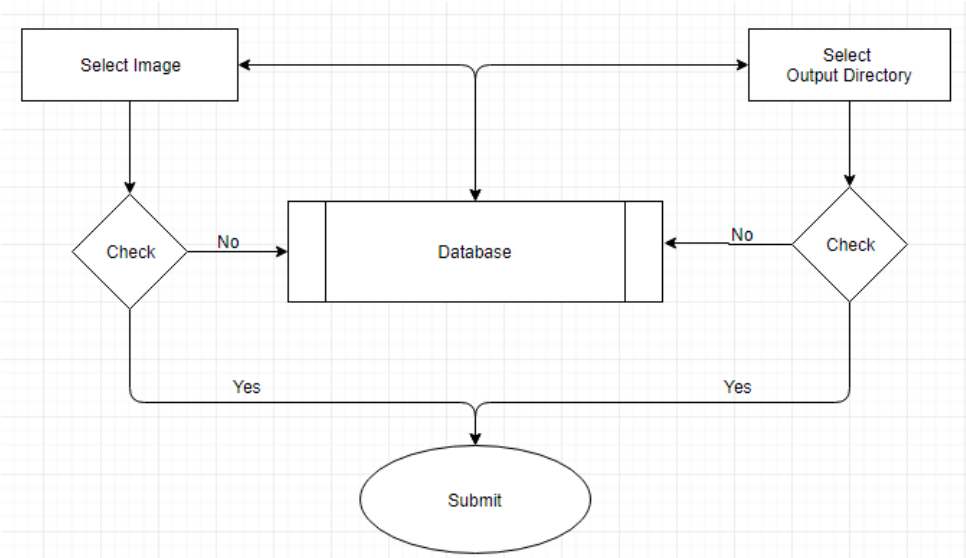
Design notation used in DFD are:



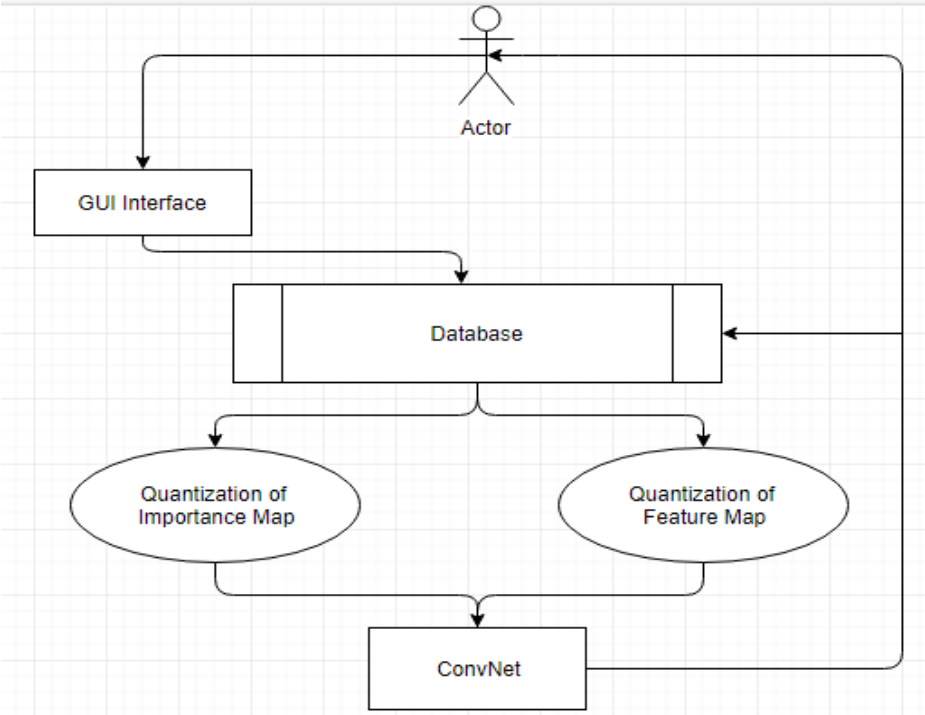
6.3. Detail Diagram

This part tells about the way the whole system works and it is explained using E-R diagrams and data flow diagram.

6.3.1. E-R Diagram

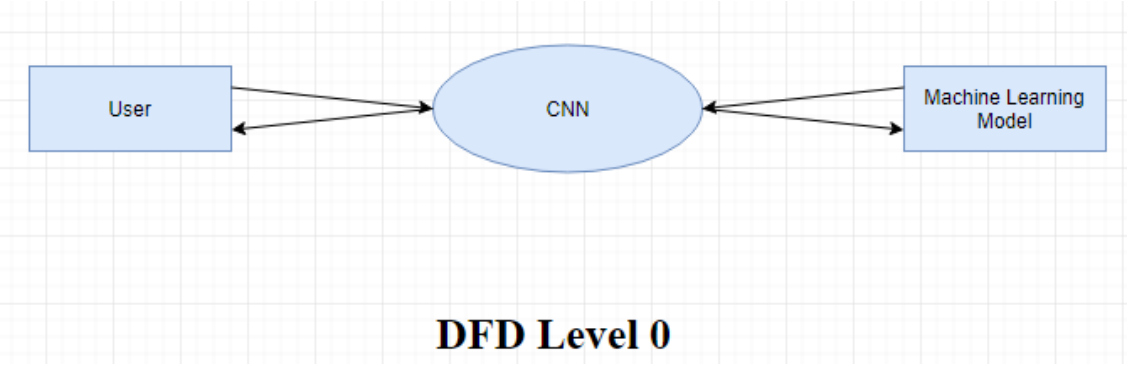


App E-R Diagram

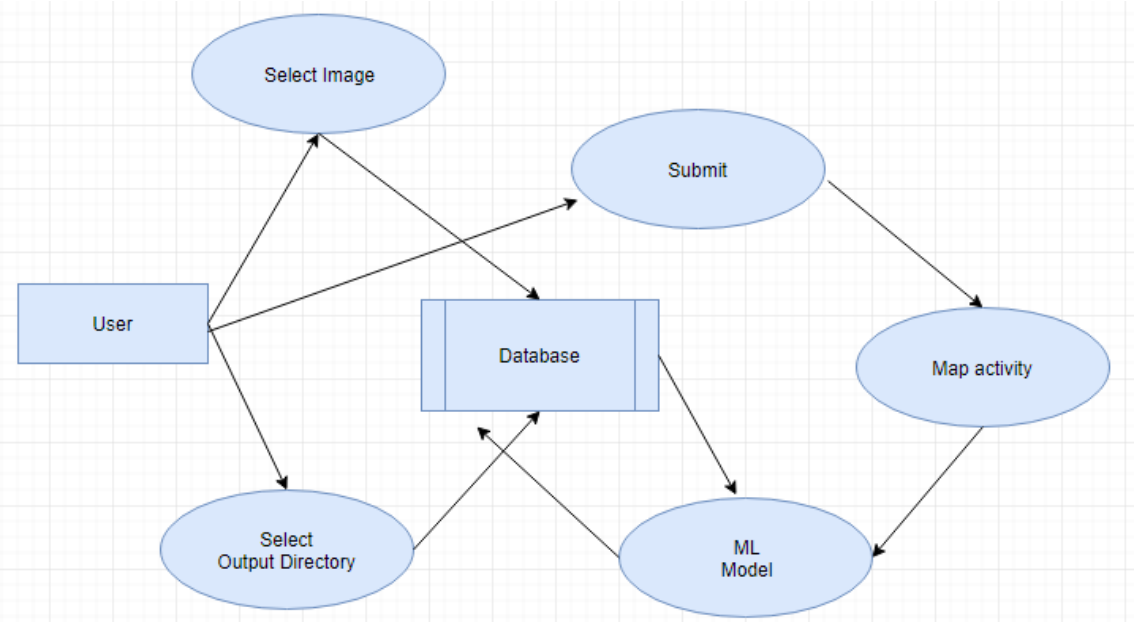


Machine Learning E-R Diagram

6.3.2. DFD's

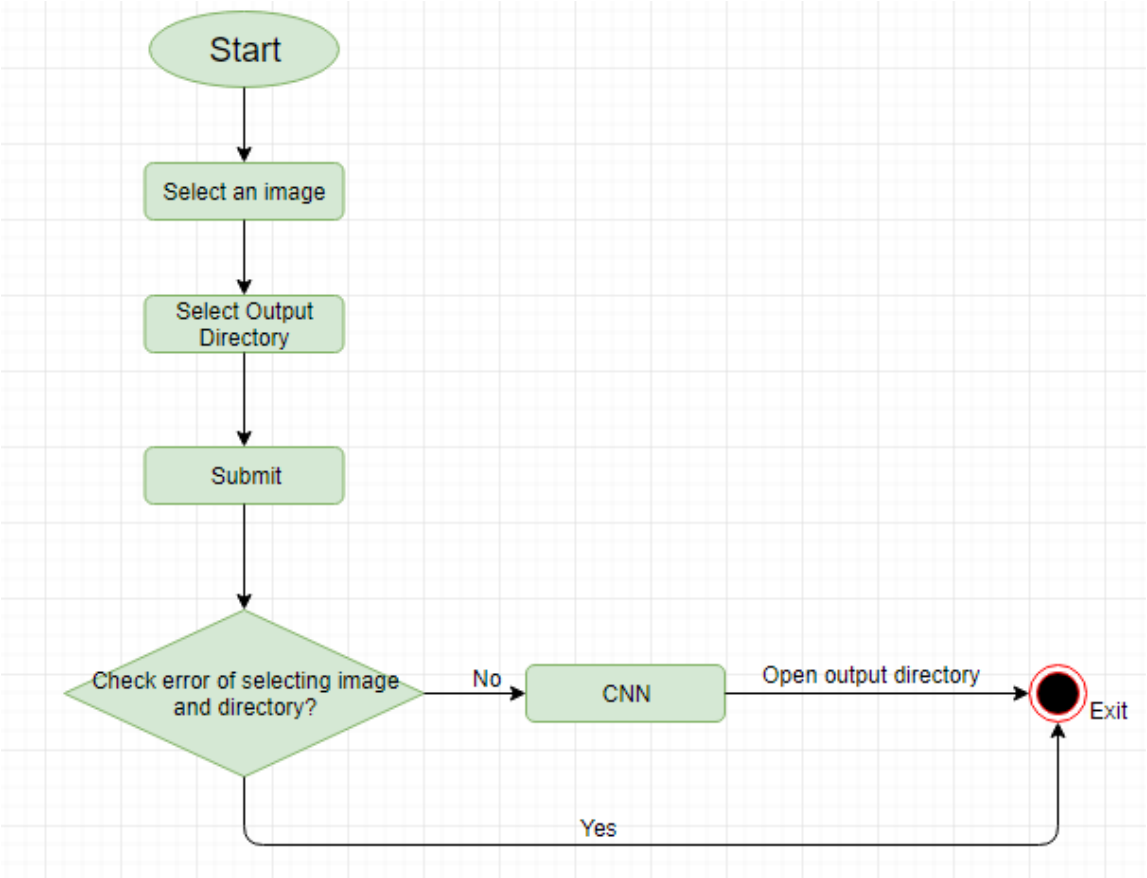


DFD Level 0



DFD Level 1

6.4. Flowcharts



7. Testing

7.1. Introduction

The software engineering can be viewed as a spiral. Initially system engineering describes the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inside along the spiral, we come to design and finally to coding. To develop GUI and CNN Model, we spiral in along rationalizes that reduce level of abstraction on each turn.

A plan for software testing may also be seen in the context of the spiral. Unit testing initiates at the vertex of the spiral and focuses on each unit of the software as implemented in source code. Testing growths by moving outward along the spiral to integration testing where the concentration is on the design and the construction of the software architecture. Talking another turn on outward on the spiral, we come across validation of image file types analysis are validated against the software that has been

constructed. Finally, we come to a system testing, where the software and other system elements are tested as a whole.

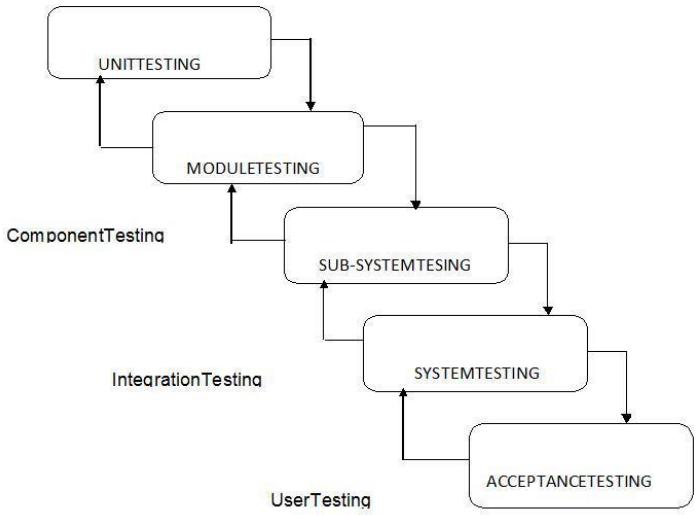


Fig 7.1. Types of testing

7.2. Functional Testing:

Functional testing is a quality assurance (QA) procedure and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by serving them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional testing usually describes what the system does.

Functional testing typically involves five steps:

1. The identification of functions that the software is expected to perform.
2. The creation of input data based on the function's specifications.
3. The determination of output based on the function's specifications.
4. The execution of the test case.
5. The comparison of actual and expected outputs.

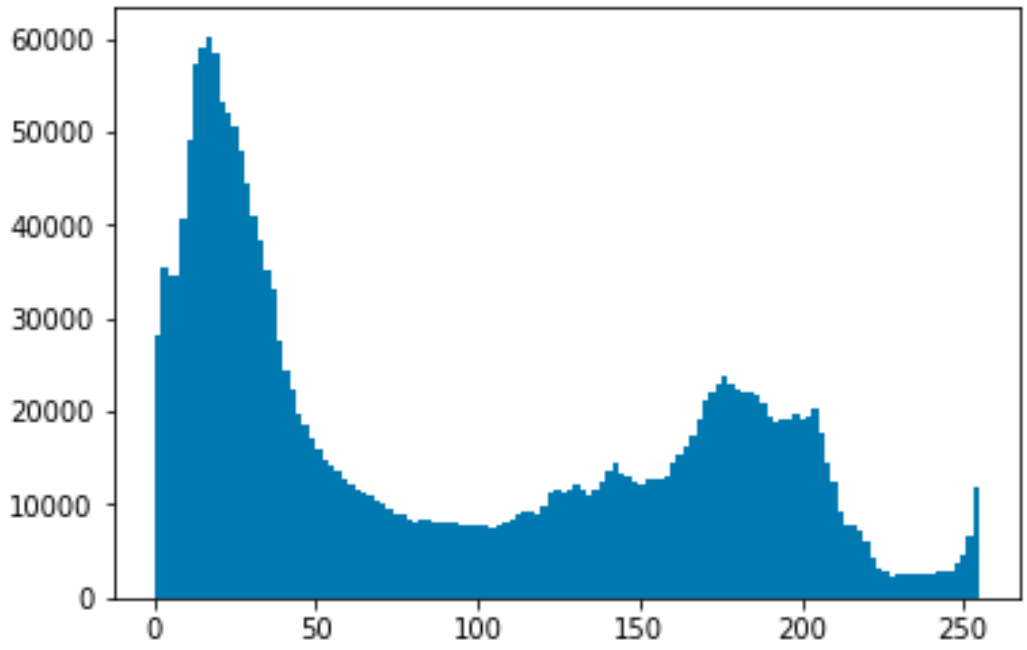
Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) devoid of looking into its internal structures

or workings (see white-box testing). This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher-level testing. Black-box testing tends to find different kinds of errors than white box testing:

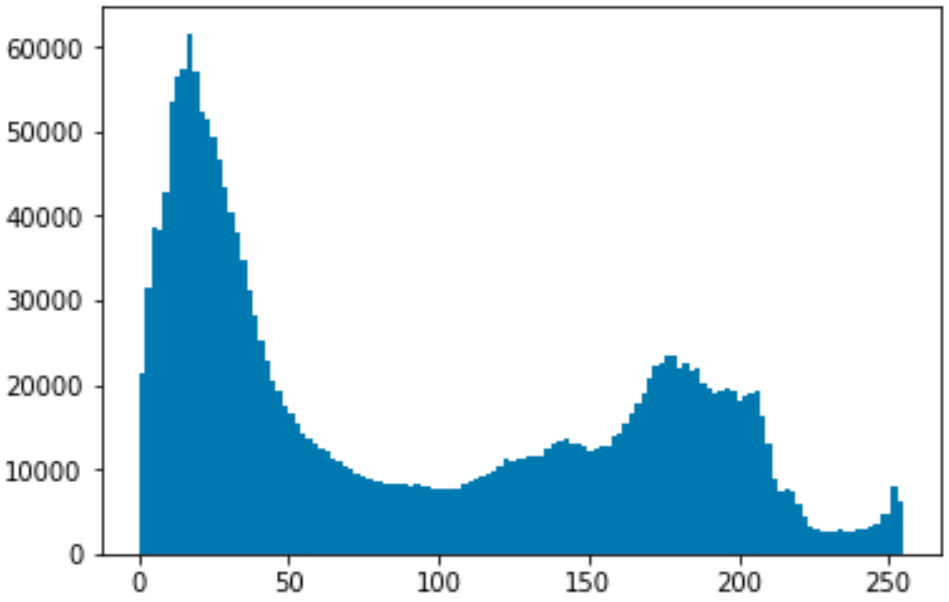
- Missing functions
- Usability problems
- Performance problem
- Concurrency and timing errors
- Initialization and termination errors

During testing, we find below graph:

Before Compression



After Compression



7.3. Structural Testing:

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests core structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing, an internal observation of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in -circuit testing (ICT).

While White-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this method of test design can uncover many errors or problems, it might not notice unimplemented parts of the specification or missing requirements.

White-box test design techniques include:

- Control flow testing
- Data flow testing
- Branch testing
- Path testing

- Statement coverage
- Decision coverage

7.4. Levels of Testing

7.4.1. Unit testing

White-box testing is done during unit testing to guarantee that the code is working as intended, before any integration take place with previously tested code. White box testing during unit testing catches any faults early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.

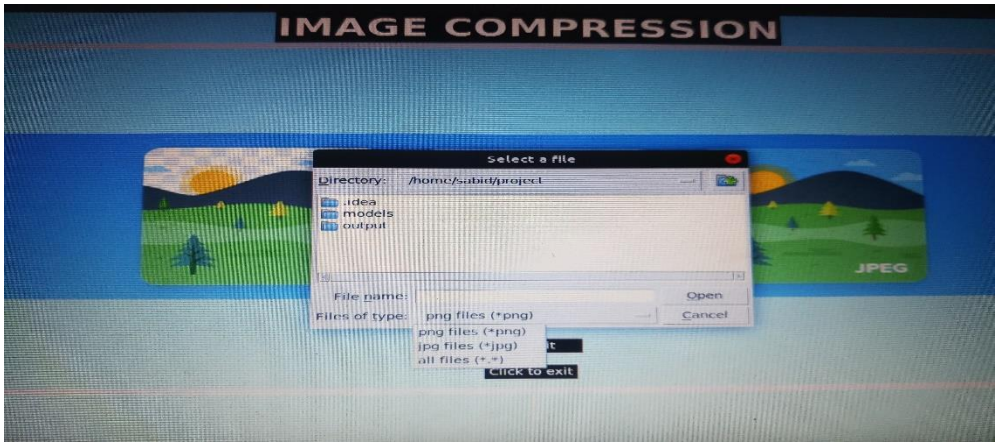
7.4.2. Integration Testing

White box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness for any interactions of interfaces that are known to the programmer.

7.5. Testing of the project

After the completion of Designing and coding of the project, we move to project testing phase. Following are the steps performed in testing phase:

1. We checked the “select image” whether only image type file is selected or not. In case we used different types of images we can select to process in algorithm to compress. If we select another file except image, then our model doesn’t process further.



2. Next, we checked the “select output directory” section whether we are going to select any directory or not for out output file.

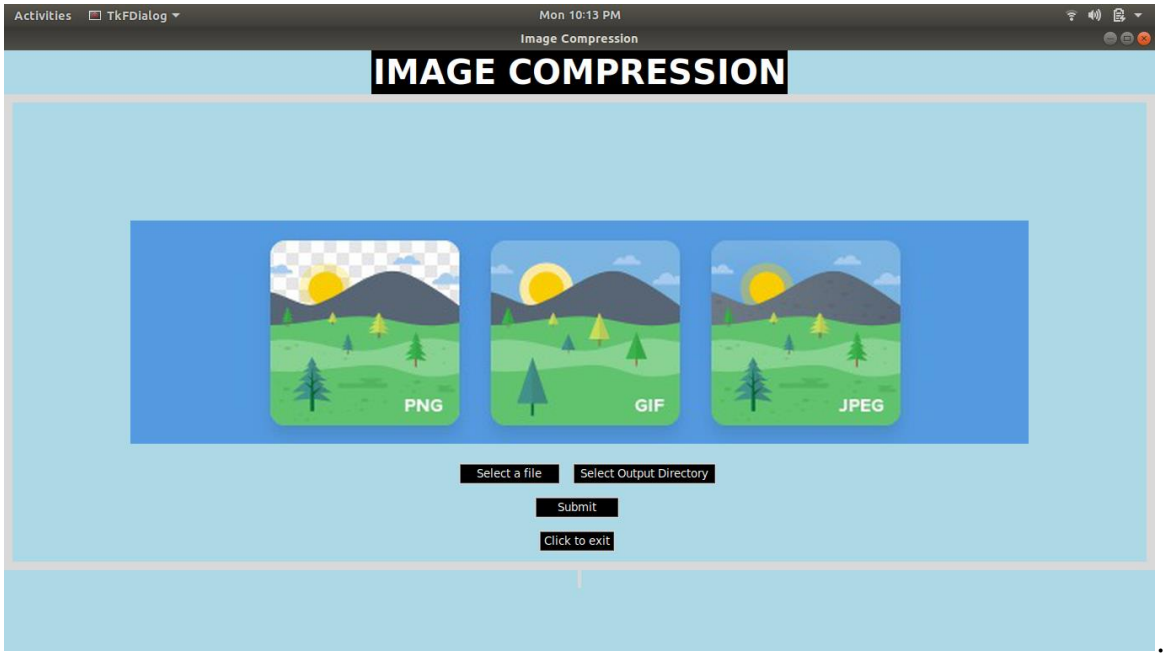
3. We then checked the “submit” section whether it is working or not. Here after clicking on submit button our model start processing the selected images and store compressed images to the selected directory or not.

4. We then checked the “Exit” section whether it is working or not. Here, we set a program that when a user clicks in it then GUI interface will close.

5. Finally, we verify the original image and compressed image with all its properties that how accurate our result is.

Original Image (Size: 1.5 MB) Compressed Image (Size: 185 KB)

Here the complete GUI is shown for user



8. Implementation:

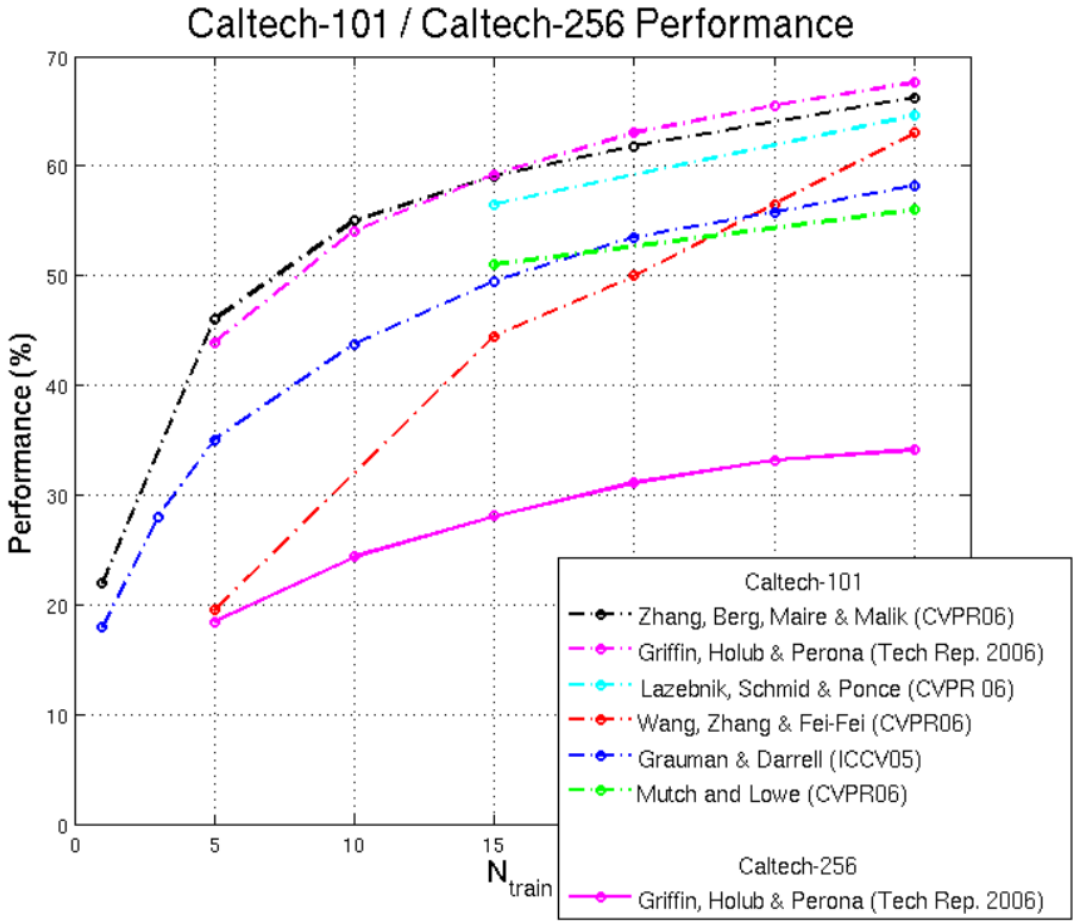
8.1. Implementation of the Project

We implemented our system using Python libraries such as NumPy, pandas, matplotlib, Tkinter, TensorFlow. These python libraries made it easy for us to handle the backend of the system where we do our main work to compress our images as much as possible by keeping their quality to maximum.

8.1.1 Implementation of Back-End

This was one of the important tasks of our project which took maximum time. For our convenience we used PyCharm community edition which help us to type our code in much more efficient manner and help us to keep our entire project on single location.

Now we get to the hard part where we must decide the image dataset to train our Machine Learning Model and we choose Caltech 256 Image Dataset which has over 30k images in 256 object categories.



Now we must prepare our data, we have over 30k images we must prepare them for both training and testing of our model and they are saved in dataset folder in pickle form where they are converted in byte stream.

After that we will train our model for the above provided train and test image dataset after which we have our model created inside model folder which we will use further in our back end.

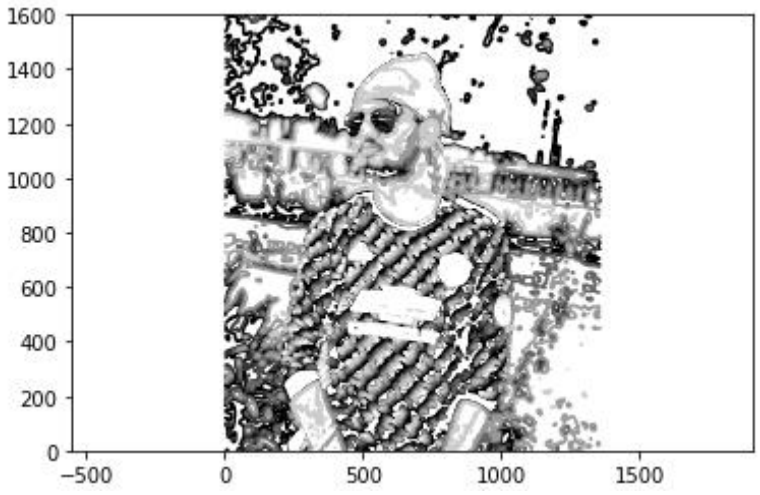


Fig: Grey Scale Image

Now in our compress file we have implemented our two modules where we create our Importance map and binary map where we use our model which we have trained to get the important feature of the images in the first part as for the second part we try to combine both Importance map (Figure a) and Binary map (Figure b) to get our compressed images.

(a). Importance Map

(b). Binary Map

8.1.1 Implementation of Front-End

Our front end is basically a simple python GUI (Graphical User Interface) using Tkinter python library

Tkinter is very useful for creation of simple GUI. It is standard python interface to the Tk GUI toolkit shipped with python. Python with Tkinter outputs the fastest and easiest way to create the GUI applications. Creating GUI using Tkinter is an easy task.

To create a Tkinter:

- Importing the module- Tkinter
- Create the main window
- Add any number of widgets to the main window
- Apply the event trigger on the widget.

where we have options to select the image file of all types like .png, .jpg etc. after which we can select the output folder where we want our image to be saved.

8.2 Post implementation and software maintenance

The focus after the implementation is to decrease the time complexity and increase our GUI interaction. As our focus is to compress the image as much as possible and to keep the image quality to its maximum state altogether.

We have trained our own model using 256 Caltech Image dataset which has different weights assigned to the images during training and testing of model.

As we are using ConvNets which is one of the main categories to do image recognition and classification etc., are some areas where CNNs are widely used. Computer sees an input image as array of pixels and it depends on the image resolution (height, width, and channels (RGB)).

For maintaining our software, one of us check it on daily basis, whether there are some changes in python library versions so that we can keep our software updated.

Our focus is to keep our coding part as simple as possible so that later if we want to add something new in our module, we don't have problem. As our group consist of members from different minor which helped us seeing our software from the different perspective which helped us in making better changes in the software.

9. Project Legacy

9.1. Current status of the project

Currently we have built the software on small scale and our GUI is hosted locally. It is not been hosted live.

After performing testing phase, we found lots of bugs in application. We fixed all the bugs after testing phase.

Our software performs well the model's problem have been solved in updated software.

9.2. Remaining Areas of concern

The remaining areas of concerns are:

- Resource consumption/ Resources required by our model of the image compression.
- Model is trained for only few epochs (epochs=5).

9.3. Technical and Managerial lessons learnt

In technical lessons, we learnt different python libraries such as NumPy, pandas, matplotlib, skimages etc.

And some of the important concepts of ConvNets, TensorFlow etc.

And on designing we learnt about python GUI Tkinter which is mostly used in python for designing simple GUIs.

Relation between human brain working of image processing and Image compression through machine learning.

In managerial lesson, we learnt about how to manage the large project by dividing them into small modules and about time management, how much time we should assign to the modules according to their priority.

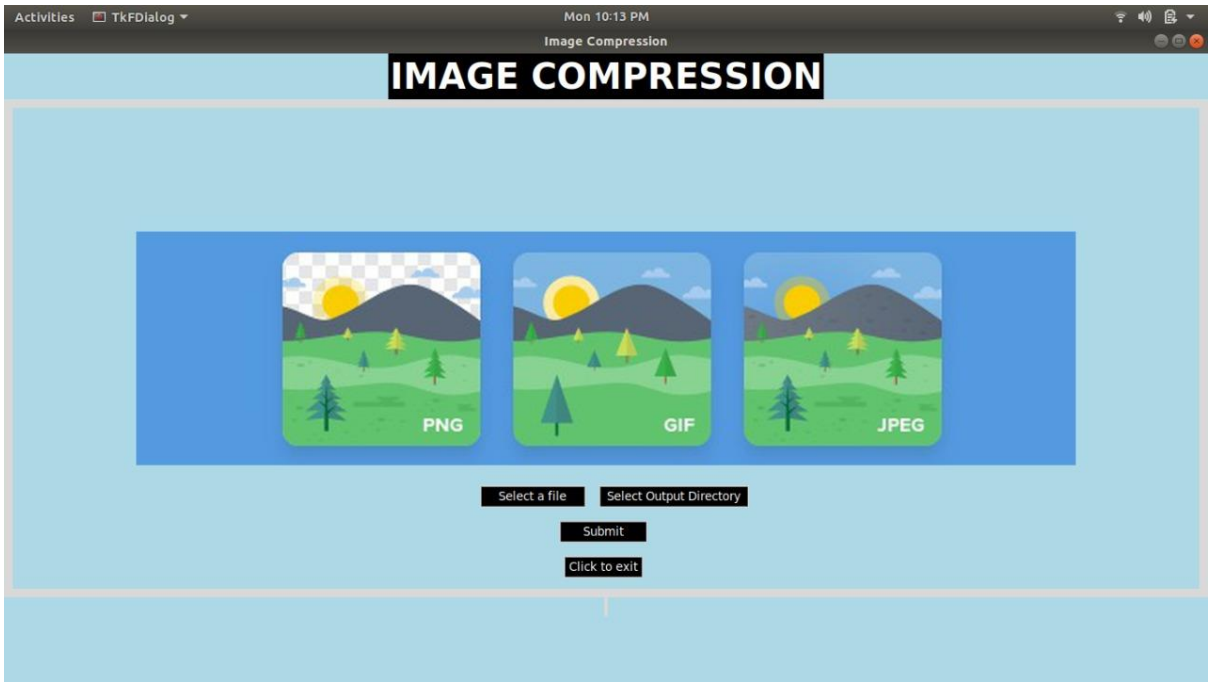
10. User Manual for Image Compression GUI

Using the Image compression GUI is very easy task.

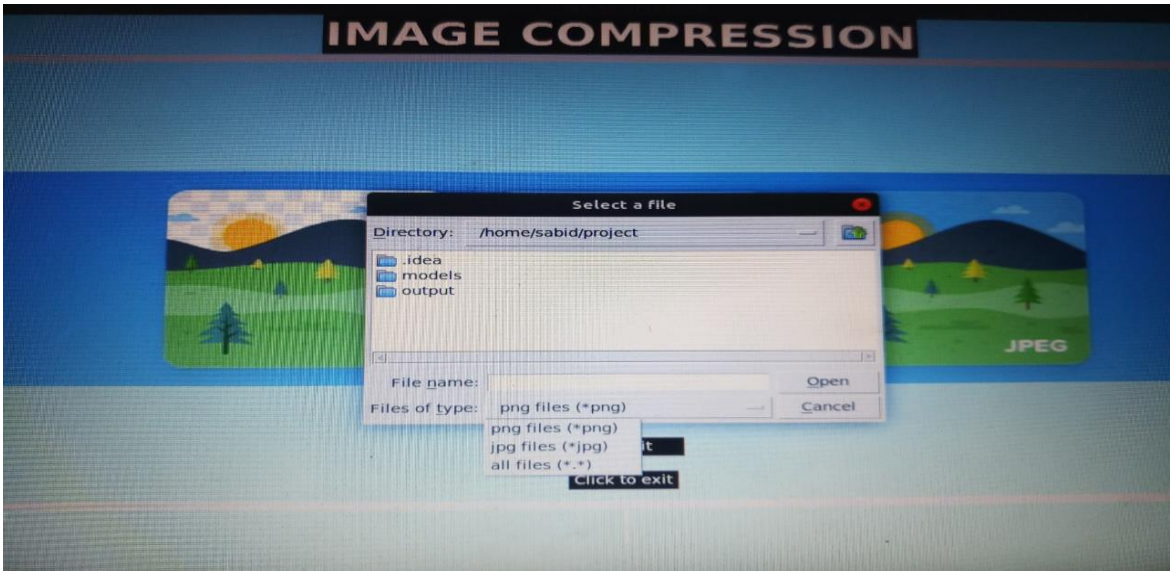
User just have to run the python code file named Compress.py and a simple GUI would appear Infront of user.

After that user have to follow the following steps for using our image compression software

Below is the image of our GUI



- After GUI appears user just have to click to the button select a file and another screen will appear to select the file which you want to compress.



- After selection of image file than user has to click on button select output directory where another screen will appear to select the directory where user want to save the compressed image.

- After all the above processes user just have to click on submit button where the code will start running on the back end and after execution of the code there will be a pop on the screen that your image has been successfully compressed.

- User compressed image will be saved on the directory where user want to save it.


Home
Desktop
Capstone_Project1
Capstone
output
original
tsm.jpg_50.
jpg

11. Source Codes

- **Main file (Compress.py)**

```
from Tkinter import *
import os
from tkFileDialog import askdirectory
import Tkinter as tk
from PIL import ImageTk, Image
import tkFileDialog as filedialog
import tkMessageBox
class CanvasButton:
   def __init__(self, canvas):
      self.canvas = canvas
      self.number = tk.IntVar()
      self.button = tk.Button(canvas, text="Select a file",bg="black",fg="white",
                        command=self.buttonclicked)
      self.id = canvas.create_window(600,450, width=120, height=25,
                              window=self.button,anchor=CENTER)
      self.button1 = tk.Button(canvas, text="Submit",bg="black",fg="white",
                        command=self.buttonclicked2)
      self.id = canvas.create_window(680, 490, width=100, height=25,
                              window=self.button1, anchor=CENTER)
      self.button2 = tk.Button(canvas, text="Click to exit", bg="black", fg="white",
                        command=self.buttonclicked3)
      self.id = canvas.create_window(680, 530, width=90, height=25,
                              window=self.button2, anchor=CENTER)
      self.button3 = tk.Button(canvas, text="Select Output Directory", bg="black",
fg="white", command=self.buttonclicked4)
      self.id = canvas.create_window(760, 450, width=170, height=25,
                              window=self.button3, anchor=CENTER)

   def buttonclicked(self):
      global my_image
      root.filename =
filedialog.askopenfilename(initialdir="/home/tarunrpmahar/Desktop", title="Select a
file", filetypes=(("png files", "*png"),("jpg files","*jpg"), ("all files", "*.*")))

my_label = Label(root, text=root.filename).pack()
      my_image = ImageTk.PhotoImage(Image.open(root.filename))
      my_image_label = Label(image=my_image).pack()

   def buttonclicked4(self):
      Tk().withdraw()
      root.dirname = filedialog.askdirectory(initialdir=os.getcwd(), title='Please select
a directory')
         print ("You chose %s" % root.dirname)
```

```
    def buttonclicked2(self):

        import os
        import sys
        import skimage.io
        import tensorflow as tf
        import matplotlib

        matplotlib.use('Agg')
        import matplotlib.pyplot as plt

        from parameters import Or_Parameter
        from set_image import ld_one_image, normalization
        from model import ConvNet

        plt.style.use('ggplot')
        image = ld_one_image(root.filename)

        good_par = Or_Parameter(verbose=False)
        tf_img = tf.compat.v1.placeholder(tf.float32, [None, good_par.image_h,
good_par.image_w, good_par.image_c],
                                          name="images")
        tf_cls = tf.compat.v1.placeholder(tf.int64, [None], name='class')

        cnn = ConvNet()
        if good_par.fine_tuning:
            cnn.ld_vgg_wts()

        last_cnn, space, P_cls = cnn.cnn_build(tf_img)
        binary_map = cnn.get_binary(tf_cls, last_cnn)

        with tf.compat.v1.Session() as sess:
            tf.compat.v1.train.Saver().restore(sess, good_par.model_path)
            last_cnn_value, P_cls_value = sess.run([last_cnn, P_cls], feed_dict={tf_img:
image})

            all_binary_pred = P_cls_value.argsort(axis=1)

            msroi = None
            for i in range(-1 * good_par.top_k, 0):

                cur_cls = all_binary_pred[:, i]
                binary_val = sess.run(binary_map, feed_dict={tf_cls: cur_cls, last_cnn:
last_cnn_value})
                norm_binary_map = normalization(binary_val[0])
```

```
            if msroi is None:
                msroi = 1.2 * norm_binary_map
            else:
                msroi = (msroi + norm_binary_map) / 2
        msroi = normalization(msroi)

    sess.close()

    figure, axis = plt.subplots(1, 1, figsize=(12, 9))
    axis.margins(0)
    plt.axis('off')
    plt.imshow(msroi, cmap=plt.cm.jet, interpolation='nearest')
    plt.imshow(image[0], alpha=0.4)

    if not os.path.exists('output'):
        os.makedirs('output')
    plt.savefig(root.dirname + '/output/Importance_map.png')
    skimage.io.imsave(root.dirname + '/output/msroi_binary_map.jpg', msroi)
    print("Both binary_map and Importance_map are created.")
#-------------------------------------------------------------------------------
    import numpy as np
    import argparse
    from PIL import Image
    import os

    arg_pars = argparse.ArgumentParser()

    arg_pars.add_argument('-jpeg_compression', type=int, default=50)
    arg_pars.add_argument('-model', type=int, default=1)
    arg_pars.add_argument('-single', type=int, default=1)
    arg_pars.add_argument('-print_metrics', type=int, default=0)
    arg_pars.add_argument('-image', type=str, default=root.filename)
    arg_pars.add_argument('-map', type=str,
default='/home/tarunrpmahar/Desktop/Capstone/output/msroi_binary_map.jpg')
    arg_pars.add_argument('-output_directory', type=str,
default='/home/tarunrpmahar/Desktop/output')
    arg_pars.add_argument('-use_convert', type=int, default=0)

    args = arg_pars.parse_args()

    def qual_compr(native, new_sal):
        if args.print_metrics:
            print (args.image)

        if native.size != new_sal.size:
```

```
            new_sal = new_sal.resize(native.size)

        new_sal_arr = np.asarray(new_sal)
        qual_img = []
        qual_pace = [i * 10 for i in range(1, 11)]

        os.makedirs('chintan')
        for q in qual_pace:
            name = 'chintan/temp_' + str(q) + '.jpg'
            if args.use_convert:
                os.system('lets see ' + str(q) + ' ' + args.image + ' ' + name)
            else:
                native.save(name, quality=q)
            qual_img.append(np.asarray(Image.open(name)))
            os.remove(name)
        os.rmdir('chintan')

        k_hare = qual_img[-1][:]
        shape = k_hare.shape
        k_hare.flags.writeable = True
        q_inp = [np.percentile(new_sal_arr, j) for j in qual_pace]
        lower, medium, higher = 1, 5, 9

        for i in range(shape[0]):
            for j in range(shape[1]):
                for k in range(shape[2]):
                    new_ss = new_sal_arr[i, j]

                    if args.model == 1:
                        for indx, q_i in enumerate(q_inp):
                            if new_ss < q_i:
                                new_qq = indx + 1
                                break

                    else:
                        raise Exception("unknown model number")

                    if new_qq < lower: new_qq = lower
                    if new_qq > higher: new_qq = higher
                    k_hare[i, j, k] = qual_img[new_qq][i, j, k]

        cmprs = args.output_directory + '/' + 'Compressed_' + args.image.split('/')[-1]
+ '_' + str(
            args.jpeg_compression) + '.jpg'
        native.save(cmprs, quality=args.jpeg_compression)
        print('compressed image in your working directory')
```

```
        if not os.path.exists(args.output_directory):
            os.makedirs(args.output_directory)

        if args.single:
            original = Image.open(args.image)
            sal = Image.open(args.map)
            qual_compr(original, sal)

        tkMessageBox.showinfo("Title", "compressed image is in desktop output
directory")

    def buttonclicked3(self):
        root.destroy()
        exit()


root = tk.Tk()
root.title("Image Compression")
root.configure(background="light blue")
theLabel=Label(root,text="IMAGE
COMPRESSION",bg="black",fg="white",font="none 30 bold")
theLabel.pack()
canvas=Canvas (width=1024, height=756)
imgpath = 'abb.jpg'
img = Image.open(imgpath)
img=img.resize((1290,662),Image.ANTIALIAS)
photo = ImageTk.PhotoImage(img)

canvas = tk.Canvas(root, bd=140, highlightthickness=10,width=3340,height=1440)
canvas.configure(background="light blue")
canvas.pack()
canvas.create_image(690, 342, image=photo)

CanvasButton(canvas)

root.mainloop()
```

- **Our CNN model code snippet:**

```
def cnn_build(self, image):

    image = self.img_cnvrsn_scaling(image)

    conv1_1 = self.conv_dep(image, "conv1_1", nonlinearity=tf.nn.relu)
    conv1_2 = self.conv_dep(conv1_1, "conv1_2", nonlinearity=tf.nn.relu)
    pool1 = tf.nn.max_pool2d(conv1_2,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool1')

    conv2_1 = self.conv_dep(pool1, "conv2_1", nonlinearity=tf.nn.relu)
    conv2_2 = self.conv_dep(conv2_1, "conv2_2", nonlinearity=tf.nn.relu)
    pool2 = tf.nn.max_pool2d(conv2_2,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool2')

    conv3_1 = self.conv_dep(pool2, "conv3_1", nonlinearity=tf.nn.relu)
    conv3_2 = self.conv_dep(conv3_1, "conv3_2", nonlinearity=tf.nn.relu)
    conv3_3 = self.conv_dep(conv3_2, "conv3_3", nonlinearity=tf.nn.relu)
    pool3 = tf.nn.max_pool2d(conv3_3,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool3')

    conv4_1 = self.conv_dep(pool3, "conv4_1", nonlinearity=tf.nn.relu)
    conv4_2 = self.conv_dep(conv4_1, "conv4_2", nonlinearity=tf.nn.relu)
    conv4_3 = self.conv_dep(conv4_2, "conv4_3", nonlinearity=tf.nn.relu)
    pool4 = tf.nn.max_pool2d(conv4_3,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool4')

    conv5_1 = self.conv_dep(pool4, "conv5_1", nonlinearity=tf.nn.relu)
    conv5_2 = self.conv_dep(conv5_1, "conv5_2", nonlinearity=tf.nn.relu)
    conv5_3 = self.conv_dep(conv5_2, "conv5_3", nonlinearity=tf.nn.relu)

    conv_depth_1 = self.conv_dep(conv5_3, "conv6_1")
    conv_depth = self.conv_dep(conv_depth_1, "depth")
    last_cnn = self.conv_dep(conv_depth, "conv6")
    space = tf.reduce_mean(last_cnn, [1, 2])

    with tf.compat.v1.variable_scope("GAP"):
```

```
        space_w = tf.compat.v1.get_variable("W",
shape=cnn_param.layer_shapes['GAP/W'],

initializer=tf.random_normal_initializer(stddev=good_par.std_dev))
    P_cls = tf.matmul(space, space_w)

    return last_cnn, space, P_cls
```

- **Preparation of data:**

```
import os
import pandas as pd
import numpy as np
from parameters import Training_Parameter

tparam = Training_Parameter(verbose=False)

image_dir_list = os.listdir(tparam.images)

label_pairs = map(lambda x: x.split('.'), image_dir_list)
labels, label_names = zip(*label_pairs)
labels = map(lambda x: int(x), labels)

label_dict = pd.Series(labels, index=label_names) - 1
image_paths_per_label = map(lambda one_dir: map(lambda one_file:
os.path.join(tparam.images, one_dir, one_file),
                                    os.listdir(os.path.join(tparam.images, one_dir))),
image_dir_list)
image_paths_train = np.hstack(map(lambda one_class: one_class[:-10],
image_paths_per_label))

image_paths_test = np.hstack(map(lambda one_class: one_class[-10:],
image_paths_per_label))

trainset = pd.DataFrame({'image_path': image_paths_train})
testset = pd.DataFrame({'image_path': image_paths_test})
print(trainset)
trainset = trainset[trainset['image_path'].map(lambda x: x.endswith('.jpg'))]
trainset['label'] = trainset['image_path'].map(lambda x: int(x.split('/')[-2].split('.')[0]) -
1)
trainset['label_name'] = trainset['image_path'].map(lambda x: x.split('/')[-
2].split('.')[1])

testset = testset[testset['image_path'].map(lambda x: x.endswith('.jpg'))]
testset['label'] = testset['image_path'].map(lambda x: int(x.split('/')[-2].split('.')[0]) - 1)
testset['label_name'] = testset['image_path'].map(lambda x: x.split('/')[-2].split('.')[1])
```

```
if not os.path.exists(tparam.data_train_path.split('/')[0]):
    os.makedirs(tparam.data_train_path.split('/')[0])

trainset.to_pickle(tparam.data_train_path)
testset.to_pickle(tparam.data_test_path)
```

- **Training of the Model:**

```
from __future__ import division
from __future__ import print_function
import math
import tensorflow as tf
import numpy as np
import pandas as pd
from time import time

from model import ConvNet
from set_image import load_img, merge
from parameters import Training_Parameter, Or_Parameter, CNN_Parameter

train_param = Training_Parameter(verbose=False)
hyper_param = Or_Parameter(verbose=False)
cparam = CNN_Parameter(verbose=False)

data_train = pd.read_pickle(train_param.data_train_path)
data_test = pd.read_pickle(train_param.data_test_path)
len_train = len(data_train)
len_test = len(data_train)
train_b_num = int(math.ceil(len_train / train_param.batch_size))
test_b_num = int(math.ceil(len_train / train_param.batch_size))

tf_images = tf.compat.v1.placeholder(tf.float32, [None, hyper_param.image_h,
hyper_param.image_w, hyper_param.image_c], name="images")
if hyper_param.empty:
    tf_labels = tf.compat.v1.placeholder(tf.int64, [None], name='labels')
else:
    tf_labels = tf.compat.v1.placeholder(tf.int64, [None, hyper_param.n_labels],
name='labels')

cnn = ConvNet()
if hyper_param.fine_tuning:
    cnn.ld_vgg_wts()

_, _, tf_prob = cnn.cnn_build(tf_images)
if hyper_param.empty:
```

```
    tf_loss =
tf.reduce_mean(tf.nn.sparse_softmax_cross_entropy_with_logits(labels=tf_labels,
logits=tf_prob))
else:
    tf_loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(tf_prob,
tf_labels))
train_loss = tf.compat.v1.summary.scalar("training loss", tf_loss)
test_loss = tf.compat.v1.summary.scalar("validation loss", tf_loss)

optimizer = tf.compat.v1.train.AdamOptimizer(train_param.learning_rate,
epsilon=0.1)

train_opt = optimizer.minimize(tf_loss)


def sparse_labels_or_not(batch):
    if hyper_param.empty:
        return batch['label'].values
    else:
        labels = np.zeros((len(batch), hyper_param.n_labels))
        for i, j in enumerate(batch['label'].values):
            labels[i, j] = 1
        return labels


with tf.compat.v1.Session() as sess:
    saver = tf.compat.v1.train.Saver()
    sess.run(tf.initialize_all_variables())

    if train_param.resume_training:
        saver.restore(sess, train_param.model_path + 'model')
        if train_param.on_resume_fix_lr:
            optimizer = tf.train.FtrlOptimizer(train_param.learning_rate)
            train_opt = optimizer.minimize(tf_loss)
        print("restore")

    summary_writer = tf.compat.v1.summary.FileWriter('tensorboards', sess.graph)

    for epoch in xrange(train_param.num_epochs):

        start = time()

        epoch_loss = 0
        for b, batch_train in enumerate(merge(data_train.sample(frac=1),
train_param.batch_size)):
```

```
            train_img = np.array(map(lambda i: load_img(i),
batch_train['image_path'].values))
            train_lab = sparse_labels_or_not(batch_train)
            _, batch_loss, sw_loss = sess.run([train_opt, tf_loss, train_loss],
                                              feed_dict={tf_images: train_img, tf_labels: train_lab})

            average_batch_loss = np.average(batch_loss)
            epoch_loss += average_batch_loss
            summary_writer.add_summary(sw_loss, epoch * train_b_num + b)
            print("Training: epoch:{}, batch:{}/{}, loss:{}".format(epoch, b, train_b_num,
average_batch_loss))
        print("Trained: epoch:{}, total loss:{}".format(epoch, epoch_loss /
train_b_num))

        # Validation
        validation_loss = 0
        for b, batch_test in enumerate(merge(data_test, train_param.batch_size)):  # no
need to randomize test batch
            test_img = np.array(map(lambda i: load_img(i),
batch_test['image_path'].values))
            test_labl = sparse_labels_or_not(batch_test)
            batch_loss, sw_loss = sess.run([tf_loss, test_loss],
                                           feed_dict={tf_images: test_img, tf_labels: test_labl})
            summary_writer.add_summary(sw_loss, epoch * test_b_num + b)
        print("Testing: epoch:{}, total loss:{}".format(epoch, validation_loss / b))
        print("Time for one epoch:{}".format(time() - start))
        saver.save(sess, train_param.model_path + '/wt_Model')

sess.close()
```

- **Other deciding parameters for Program**

```
import pickle as cPickle
from parameters import CNN_Parameter, Or_Parameter
import tensorflow as tf

good_par = Or_Parameter(verbose=False)
cnn_param = CNN_Parameter(verbose=False)


class ConvNet():

    def p(self, t):
        print(t.name, t.get_shape())

    def ld_vgg_wts(self):
```

```
        with open(good_par.vgg_weights, "rb") as f:
            self.pretrained_weights = cPickle.load(f, fix_imports=True, errors="strict")

    def cnn_build(self, image):

        image = self.img_cnvrsn_scaling(image)

        conv1_1 = self.conv_dep(image, "conv1_1", nonlinearity=tf.nn.relu)
        conv1_2 = self.conv_dep(conv1_1, "conv1_2", nonlinearity=tf.nn.relu)
        pool1 = tf.nn.max_pool2d(conv1_2,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool1')

        conv2_1 = self.conv_dep(pool1, "conv2_1", nonlinearity=tf.nn.relu)
        conv2_2 = self.conv_dep(conv2_1, "conv2_2", nonlinearity=tf.nn.relu)
        pool2 = tf.nn.max_pool2d(conv2_2,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool2')

        conv3_1 = self.conv_dep(pool2, "conv3_1", nonlinearity=tf.nn.relu)
        conv3_2 = self.conv_dep(conv3_1, "conv3_2", nonlinearity=tf.nn.relu)
        conv3_3 = self.conv_dep(conv3_2, "conv3_3", nonlinearity=tf.nn.relu)
        pool3 = tf.nn.max_pool2d(conv3_3,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool3')

        conv4_1 = self.conv_dep(pool3, "conv4_1", nonlinearity=tf.nn.relu)
        conv4_2 = self.conv_dep(conv4_1, "conv4_2", nonlinearity=tf.nn.relu)
        conv4_3 = self.conv_dep(conv4_2, "conv4_3", nonlinearity=tf.nn.relu)
        pool4 = tf.nn.max_pool2d(conv4_3,
ksize=cnn_param.pool_window,strides=cnn_param.pool_stride, padding='SAME',
name='pool4')

        conv5_1 = self.conv_dep(pool4, "conv5_1", nonlinearity=tf.nn.relu)
        conv5_2 = self.conv_dep(conv5_1, "conv5_2", nonlinearity=tf.nn.relu)
        conv5_3 = self.conv_dep(conv5_2, "conv5_3", nonlinearity=tf.nn.relu)

        conv_depth_1 = self.conv_dep(conv5_3, "conv6_1")
        conv_depth = self.conv_dep(conv_depth_1, "depth")
        last_cnn = self.conv_dep(conv_depth, "conv6")
        space = tf.reduce_mean(last_cnn, [1, 2])

        with tf.compat.v1.variable_scope("GAP"):
            space_w = tf.compat.v1.get_variable("W",
shape=cnn_param.layer_shapes['GAP/W'],
```

```
initializer=tf.random_normal_initializer(stddev=good_par.std_dev))

        P_cls = tf.matmul(space, space_w)

        return last_cnn, space, P_cls

    def get_binary(self, tf_cls, last_cnn):
        with tf.compat.v1.variable_scope("GAP", reuse=True):
            class_w = tf.gather(tf.transpose(tf.compat.v1.get_variable("W")), tf_cls)
            class_w = tf.reshape(class_w, [-1, cnn_param.last_features, 1])
        last_cnn1 = tf.compat.v1.image.resize_bilinear(last_cnn, [good_par.image_h,
good_par.image_w])
        last_cnn1 = tf.reshape(last_cnn1, [-1, good_par.image_h * good_par.image_w,
cnn_param.last_features])
        binary_map = tf.reshape(tf.matmul(last_cnn1, class_w), [-1, good_par.image_h,
good_par.image_w])
        return binary_map

    def get_vgg_wts(self, layer_name, bias=False):
        layer = self.pretrained_weights[layer_name]
        if bias: return layer[1]
        return layer[0].transpose((2, 3, 1, 0))

    def conv_dep(self, input_, name, nonlinearity=None):
        with tf.compat.v1.variable_scope(name) as scope:

            W_shape = cnn_param.layer_shapes[name + '/W']
            b_shape = cnn_param.layer_shapes[name + '/b']

            if good_par.fine_tuning and name not in ['conv6', 'conv6_1', 'depth']:
                W = self.get_vgg_wts(name)
                b = self.get_vgg_wts(name, bias=True)
                W_initializer = tf.constant_initializer(W)
                b_initializer = tf.constant_initializer(b)
            else:
                W_initializer = tf.truncated_normal_initializer(stddev=good_par.std_dev)
                b_initializer = tf.constant_initializer(0.0)

            conv_wts = tf.compat.v1.get_variable("W", shape=W_shape,
initializer=W_initializer)
            conv_bias = tf.compat.v1.get_variable("b", shape=b_shape,
initializer=b_initializer)

            if name == 'depth':
```

```
            conv = tf.compat.v1.nn.depthwise_conv2d_native(input_, conv_wts, [1, 1,
1, 1], padding='SAME')
        else:
            conv = tf.nn.conv2d(input_, conv_wts, [1, 1, 1, 1], padding='SAME')

        bias = tf.nn.bias_add(conv, conv_bias)
        bias = tf.nn.dropout(bias, 0.7)
        if nonlinearity is None:
            return bias
        return nonlinearity(bias, name=name)

    def img_cnvrsn_scaling(self, image):
        image = image*255.
        r, g, b = tf.split(image, 3, 3)
        VGG_MEAN = [103.939, 116.779, 123.68]
        return tf.concat([b - VGG_MEAN[0], g - VGG_MEAN[1], r -
VGG_MEAN[2]],3)
```

12. Bibliography

1. https://www.tensorflow.org/tfx/tutorials/transform/census
2. https://www.analyticsvidhya.com/blog/2018/11/implementation-faster-r-cnn-python-object-detection/
3. https://docs.opencv.org/4.1.1/d5/de5/tutorial_py_setup_in_windows.html
4. https://scikit-image.org/docs/dev/api/skimage.html
5. https://github.com/ageron/tf2_course
6. https://heartbeat.fritz.ai/a-2019-guide-to-deep-learning-based-image-compression-2f5253b4d811
7. https://heartbeat.fritz.ai/a-2019-guide-to-deep-learning-based-image-compression-2f5253b4d811#6b94
8. https://heartbeat.fritz.ai/a-2019-guide-to-deep-learning-based-image-compression-2f5253b4d811#4796
9. https://www.dropbox.com/s/izfas78534qjg08/models.tar.gz?dl=0