A Project Report

On

Age and Gender Prediction

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

[2016-2020]

Submitted By

Rajkumar – 1605310030

Nisha – 1605310025

Rohit Patel – 1605310033

Under the supervision of

Mr. A.P. Singh

(Assistant Professor)

AZAD INSTITUTE OF ENGINEERING & TECHNOLOGY, LUCKNOW

Department Of Computer Science & Engineering

Affiliated to DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY,

Lucknow, Dec, 2016

CERTIFICATE

Certified that **Rajkumar**, **Nisha, Rohit Patel** has carried out the Project work presented in this report entitled **“AGE AND GENDER PREDECTION”**for the B.Tech. (Computer Science & Engineering) Fourth Year from **Azad Institute of Engineering & Technology, Lucknow** under my supervision. The report embodies result of original work and studies carried out by student himself/herself and the contents of the project do not form the basis for the award of any other degree to the candidate or to anybody else.

Mr. Pervez Rauf Khan
Head
Dept. of Computer Science & Engg.
Azad Institute of Engineering and Technology

Mr.-Atma Prakash Singh
(Project Co-ordinator)
Dept. of Computer Science & Engg.
AIET,Lucknow

Date:-

Table of Contents

ABSTRACT

Automatic face identification and verification from facial images attain good accuracy with large sets of training data while face attribute recognition from facial images still remain challengeable. We propose a methodology for automatic age and gender classification based on feature extraction from facial images, namely, primary and secondary features. Our methodology includes three main iterations: Pre-processing, Feature extraction and Classification. Our solution is able to classify images in different lighting conditions and different illumination conditions.

The project identifies or detects the gender from the given face images. The tools used involve Convolutional Neural Network along with programming language like Python. The project has been motivated by problems like lack of security, frauds, robbery, and criminal identification.

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people. I would like to give my thanks to **Mr. Pervez Rauf Khan** (Head of Department), Department of Computer Science & Engineering, Azad Institute of Engineering & Technology, Lucknow (U.P.) for providing me golden opportunity to work in such a prestigious project.

I would also like to thank my very supportive faculty, **Mr. A.P. Singh** (Assistant Professor, Azad Institute of Engineering & Technology, Lucknow (U.P.) for being my project supervisor and guiding me through the project work to complete it on the given time limit.

Rajkumar

Nisha

Rohit Patel

LIST OF FIGURES

1. INTRODUCTION

Personal monitoring, identification and verification using facial images are known to be an actively growing area of research in many computer vision applications. Some examples in this area are face recognition, face action classification, poses recognition, skin colour classification, age estimation, gender recognition and ethnicity recognition. Face recognition has achieved better results according to the research done for nearly three decades. However, similar accuracy of classification could not be gained from facial attribute recognition.

Human brain is the most powerful and accurate classifier in pattern recognition. It has the brilliant power as it is a dynamic organ involved with training and learning for a specific period of time. Our main attempt is to convert this biological and behavioural characteristic of human brain into artificial neurons in order to attain the same or better results. One of the primary uses of this work is to classify human attributes like age, gender and ethnicity using facial images.

I. Basic Idea

Vision is a web application. It is developed based on the Deep Learning system, and has computer vision enabled. The programming language used is Python. Our approach is to building fast working app which classifies the gender of a person by facial features. It includes a feature which also detects the age of the person. The application is capable of taking input in both ways i.e. through camera and also an image. A subscription feature is also introduced for the users. With this computer one can secure his/her home, verify the real age of people and allow only those who are capable. The application can be used widely in different areas. Equipped with advance features, and technology it enhance the development of Deep learning.

Our attempt with this application is to come up with an accurate method for age and gender classification from facial images. Gender classification is done according to the geometric difference of primary features in male and female.This algorithm can classify the facial images in to four age groups 0-5, 8-13, 14-25, 26-45 and 46-60. Age classification is based on the texture variation of wrinkle density in the forehead, eye lids and cheek area. Classification is done using two separate neural networks for age and gender. The project uses Deep Learning Technology where Convolutional Neural Network (CNN) acts as a classifier. CNN is used in applications where both classification speed and maximum accuracy is considered important unlike Neural Networks which focuses on classification speed.

The application is free to use and can be used by everyone. The application is completely safe and secure for use. Basically it's an application that can be used at outdoors of movie halls, hospitals, play areas, swimming pools, segregatedschools, gyms etc. The application can also be used by website developers for verification process of the user.

The use of Deep Learning gives the application the ability to enhance itself by learning daily from live data. The application gets more accurate and effective with continuous use. This enlightens the concept of machine learning.

II. Objective

The objective of this project is to build a web app which predicts the gender and age of a person by just capturing its face or a picture using deep learning. The application can be beneficial for security and also to increase the capabilities of AI-based technology systems and applications.

1.1 Technologies Used

1.1.1 Computer Vision

Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using digital images from cameras and videos and deep learning models, machines can accurately identify and classify objects — and then react to what they "see."

1.1.2 OpenCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as Numpuy, python is capable of processing the OpenCV array structure for analysis. To identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

Applications of OpenCV: There are lots of applications which are solved using OpenCV, some of them are listed below:-

- Face recognition/ Object recognition

- Automated inspection and surveillance
- Number of people – count (foot traffic in a mall, etc.)
- Vehicle counting on highways along with their speeds
- Anatomoly (defect) detection in the manufacturing process (the odd defective products)
- Street view image stitching
- Video/image search and retrieval
- Robot and driver-less car navigation and control
- Medical image analysis
- Movies – 3d structure from motion

1.1.3 Deep Learning:-

Deep learning is an artificial intelligence function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence (AI) that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network.

Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms, and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing.

However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unravelling this wealth of information and are increasingly adapting to AI systems for automated support.

A deep learning model is designed to continually analyse data with a logic structure similar to how a human would draw conclusions. To achieve this, deep learning applications use a layered structure of algorithms called an artificial neural network. The design of an artificial neural network is inspired by the biological neural network of the human brain, leading to a process of learning that’s far more capable than that of standard machine learning models.
It’s a tricky prospect to ensure that a deep learning model doesn’t draw incorrect conclusions—like other examples of AI, it requires lots of training to get the learning processes correct. But when it works as it’s intended to, functional deep

learning is often received as a scientific marvel that many consider being the backbone of true artificial intelligence.

A great example of deep learning is Google's AlphaGo. Google created a computer program with its own neural network that learned to play the abstract board game called Go, which is known for requiring sharp intellect and intuition. By playing against professional Go players, AlphaGo's deep learning model learned how to play at a level never seen before in artificial intelligence, and did without being told when it should make a specific move (as a standard machine learning model would require). It caused quite a stir when AlphaGo defeated multiple world-renowned "masters" of the game—not only could a machine grasp the complex techniques and abstract aspects of the game, it was becoming one of the greatest players of it as well.

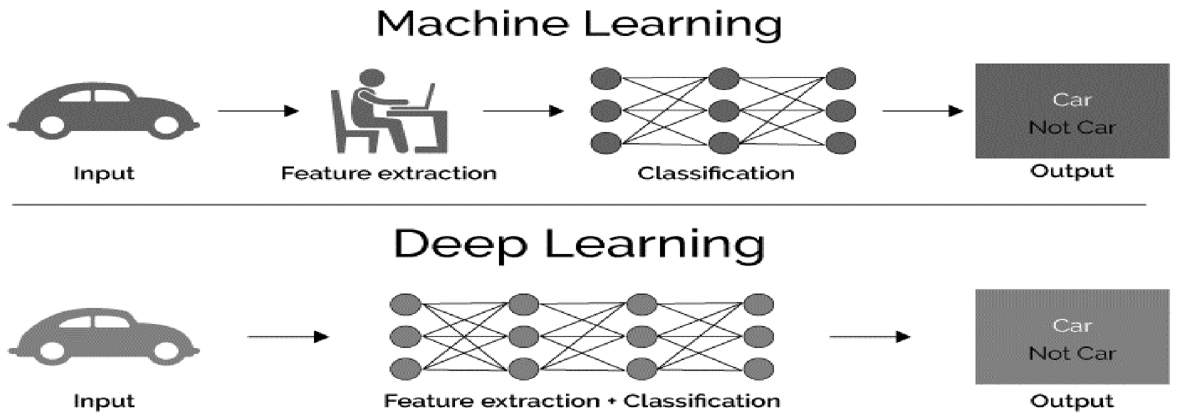


Figure 1: Difference between ML and DL

To recap the differences between the two:

- Machine learning uses algorithms to parse data, learn from that data, and make informed decisions based on what it has learned.
- Deep learning structures algorithms in layers to create an "artificial neural network" that can learn and make intelligent decisions on its own.

1.1.4 Convolutional Neural Network

Convolutional Neural Networks are a type of Deep Learning Algorithm that take the image as an input and learn the various features of the image through filters. This allows them to learn the important objects present in the image, allowing them to discern one image from the other. For example, the convolutional network will learn the specific features of cats that differentiate from the dogs so that when we provide input of cats and dogs, it can easily differentiate between the two.

One important feature of Convolutional Neural Network that sets it apart from other Machine Learning algorithms is its ability to pre-process the data by itself. Thus, you may not spend a lot of resources in data pre-processing. During cold-start, the filters may require hand engineering but with progress in training, they are able to adapt to the learned features and develop filters of their own. Therefore, CNN is continuously evolving with growth in the data.

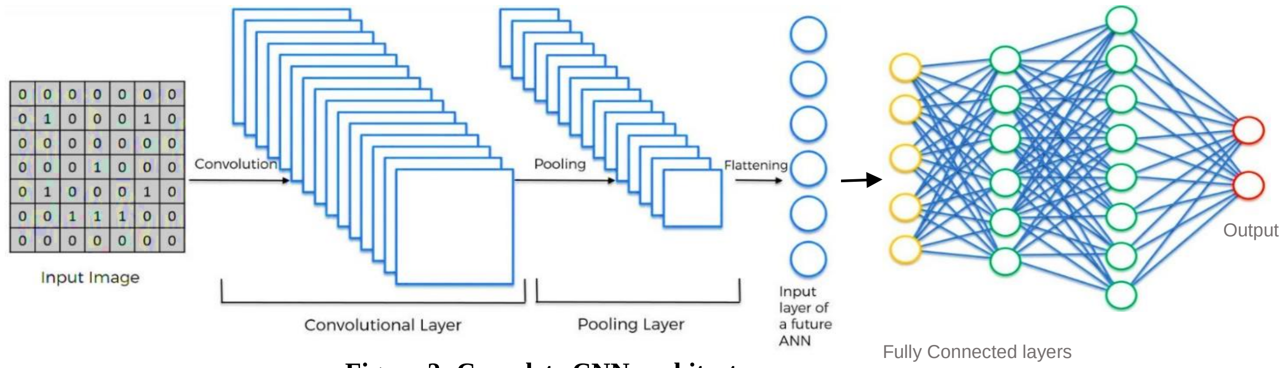


Figure 2: Complete CNN architecture

CNN image classifications takes an input image, process it and classify it under certain categories (Eg. Male, Female). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see h x w x d (h = Height, w = Width, d = Dimension). Eg, An image of 6 x 6 x 3 array of matrix of RGB (3 refers to RGB values) and an image of 4 x 4 x 1 array of matrix of grayscale image.

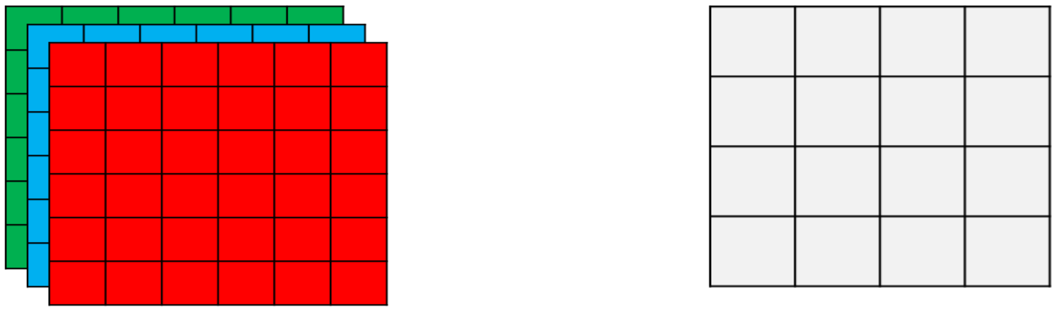


Figure 3(a): Array of rgb matrix (b): Array of grayscale

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1.

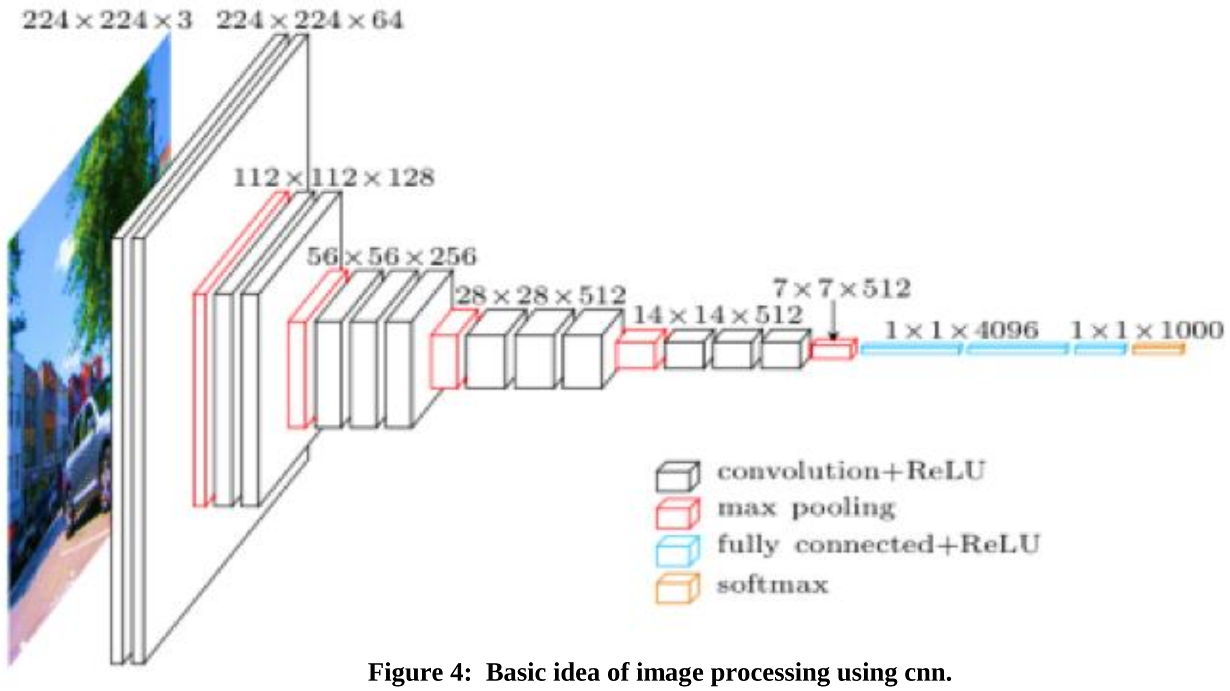


Figure 4: Basic idea of image processing using cnn.

1.1.4.1 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 kernel.

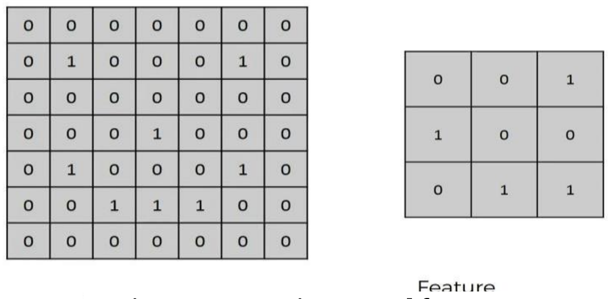


Figure 5: Input image and feature

Consider a 7 x 7 whose image pixel values are 0, 1 and filter matrix 3 x 3 as shown in above. Then the convolution of 7 x 7 image matrix multiplies with 3 x 3 filter matrix which is called **“Feature Map”** as output shown in below.

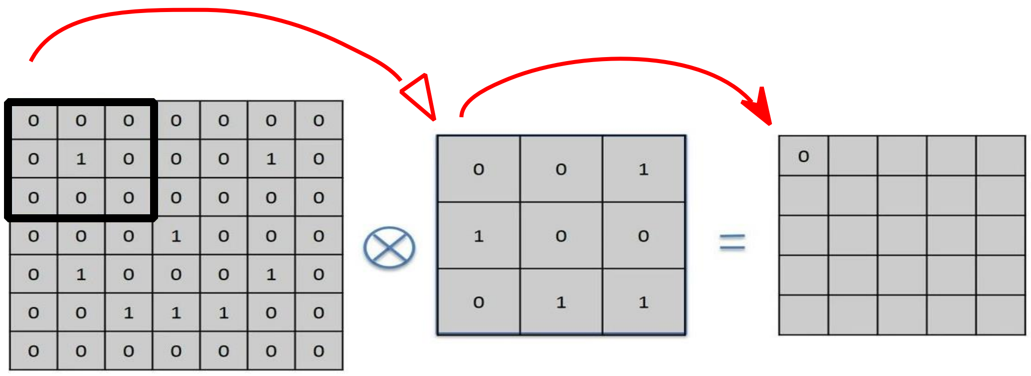
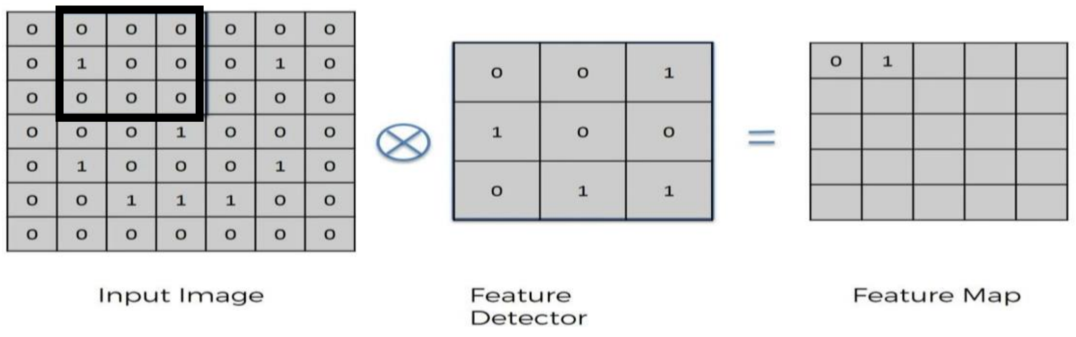
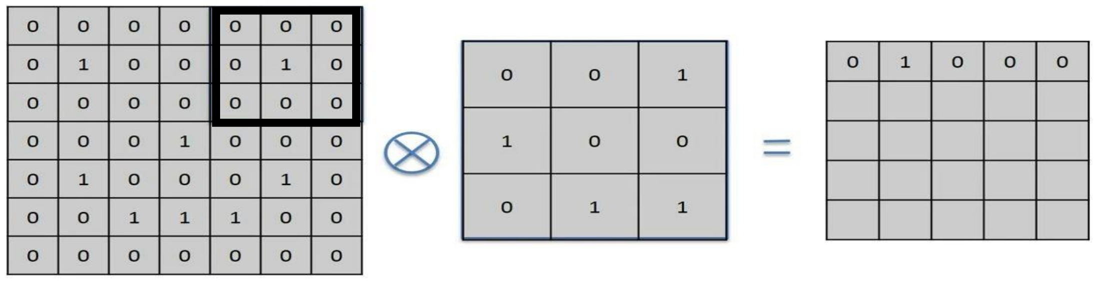
a)


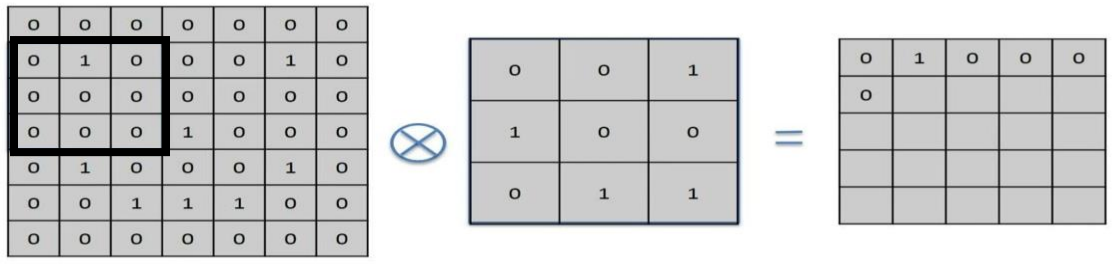
Figure 4 (a): Image matrix multiplies kernel or filter matrix



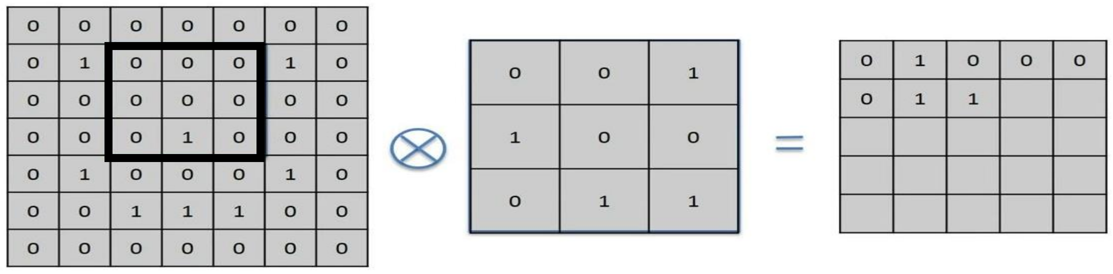
b)



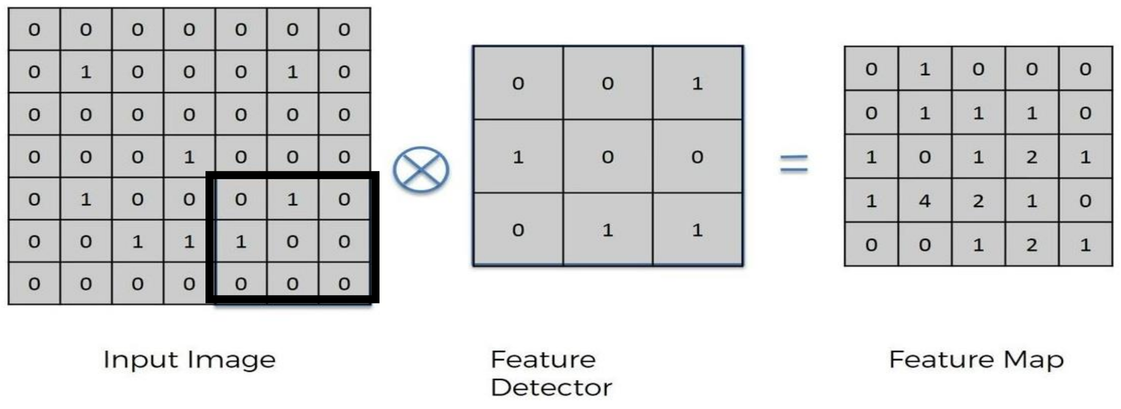
c)



d)



e)



f)

Figure 6 (f): Output Matrix

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).

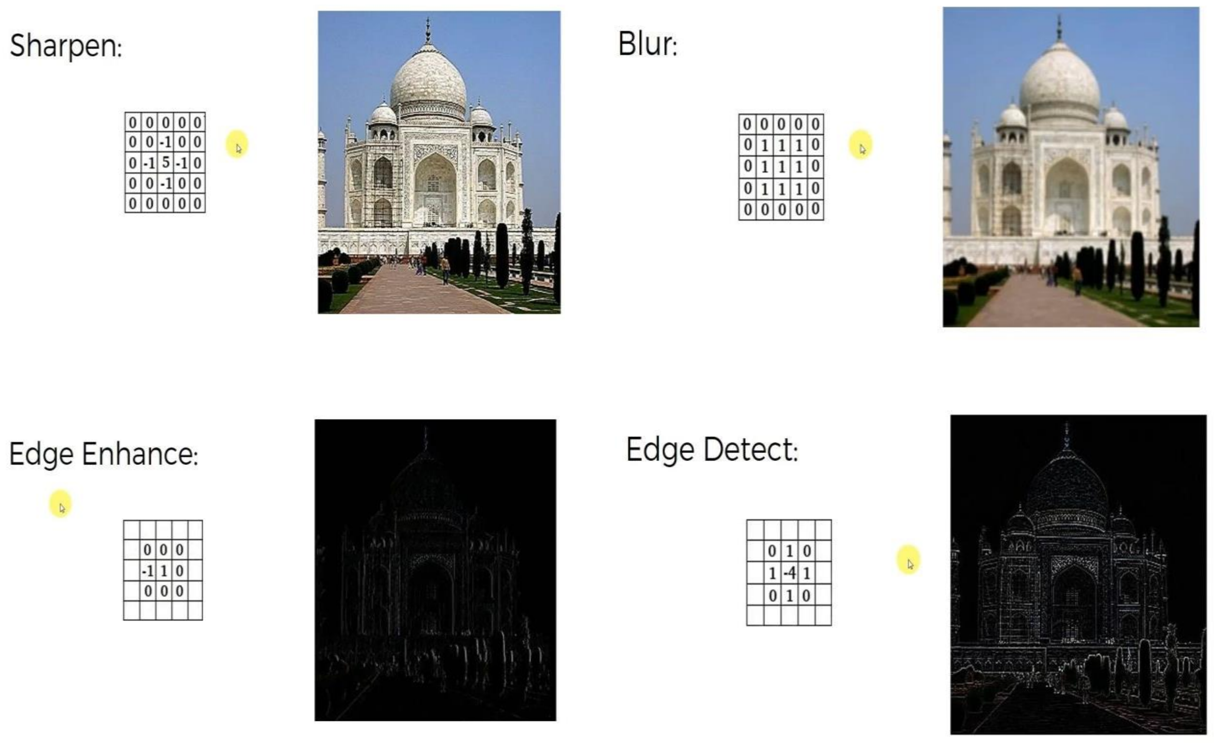


Figure 7: Some common filters

1.1.4.2 Relu Layer

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is

f(x) = max (0, x).

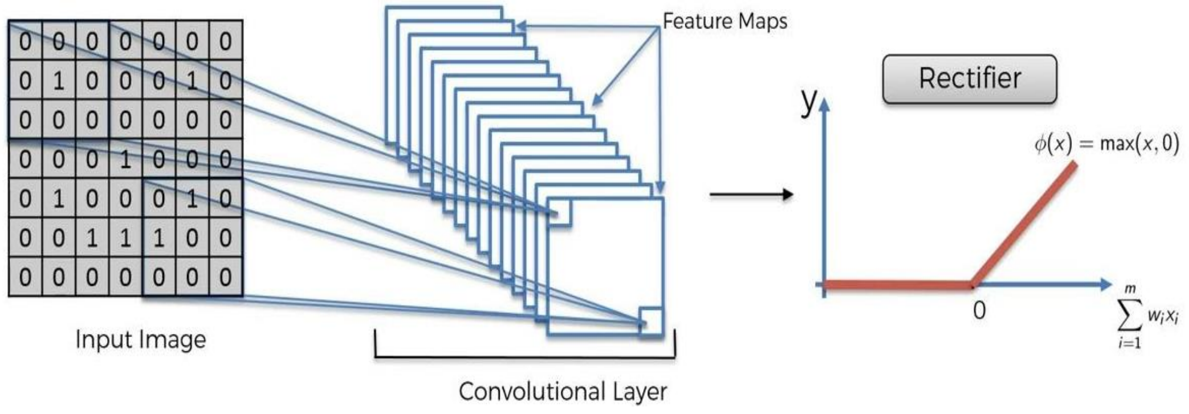


Figure 8: ReLU operation

ReLU's purpose is to introduce non-linearity in our Convolution Layer. Since, the real world data would want our Convolution Layer to learn would be non-negative linear values. Most of the data scientists use ReLU since performance wise ReLU is better than the other two.

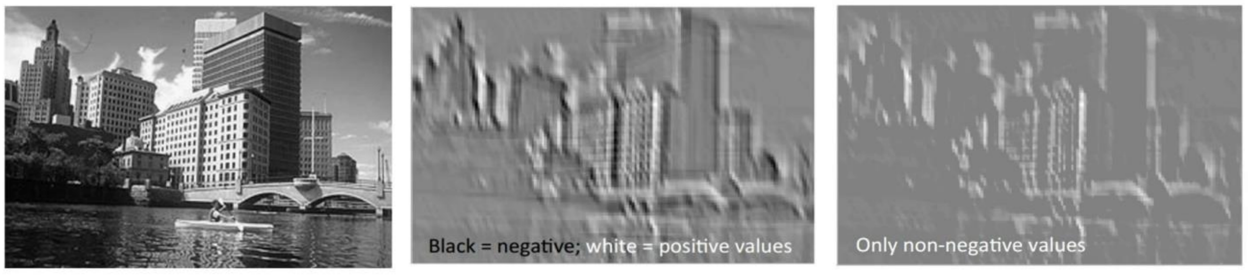


Figure 9 (a): Normal Image (b): Convolution layer output (c): ReLU layer output

1.1.4.3 Max Pooling

Pooling layers section would reduce the number of parameters when the images are too large. Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.

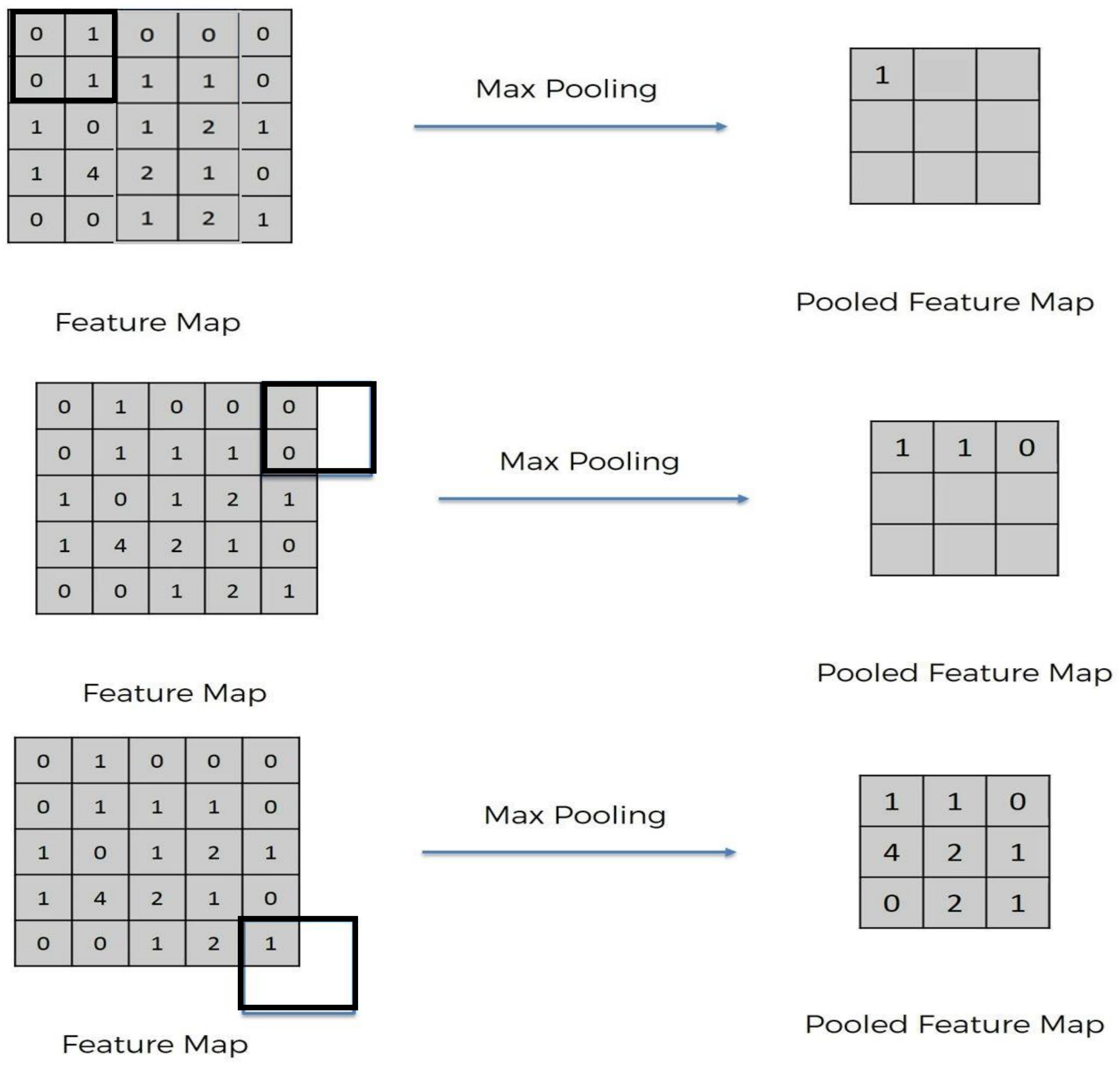
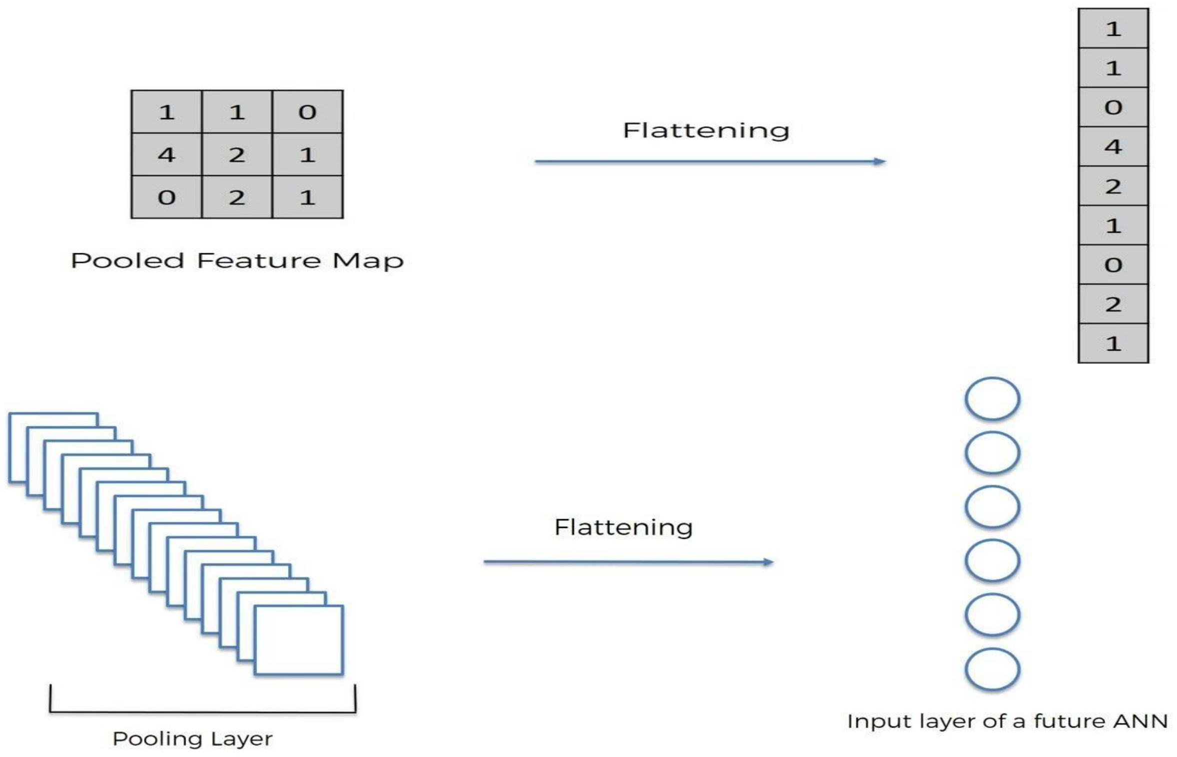


Figure 10: Max pooling output matrix

1.1.4.4 Flattening

Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. The last stage of a convolutional neural network (CNN) is a classifier. It is called a dense layer, which is just an artificial neural network (ANN) classifier. And an ANN classifier needs individual features, just like any other classifier. This means it needs a feature vector. Therefore, you need to convert the output of the convolutional part of the CNN into a 1D feature vector, to be used by the ANN part of it. This operation is called flattening. It gets the output of the convolutional layers, flattens all its structure to create a single long feature vector to be used by the dense layer for the final classification.



Adding multiple convolutional layers and pooling layers, the image will be processed for feature extraction. As the layers go deeper and deeper, the features that the model deals with become more complex. For example, at the early stage of

Conv.layer, it looks up for oriented line patterns and then finds some simple figures. At the deep stage, it can catch the specific forms of objects and finally able to detect the object of an input image.

This is an artificial neural network in cnn. An **input layer** which is the data we provide to the ANN**.** We have the **hidden layers**, which is where the magic happens. Lastly, we have the **output layer**, which is where the finished computations of the network are placed for us to use.

1.1.4.5 Fully Connected Layer

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network.In our ANNs we have these units of calculation called neurons.

These artificial neurons are connected by synapses which are really just weighted values. What this means is that given a number, a neuron will perform some sort of calculation and then the result of this calculation will be multiplied by a weight as it “travels.”

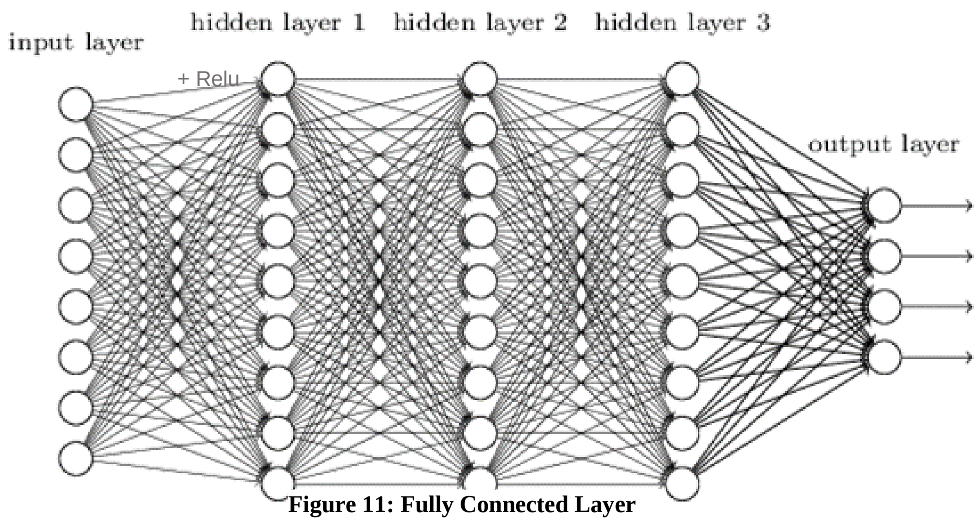


Figure 11: Fully Connected Layer

1.2 Front End

1.2.1 HTML 5

HTML5 is the next major revision of the HTML standard superseding HTML 4.01, XHTML 1.0, and XHTML 1.1. HTML5 is a standard for structuring and presenting content on the World Wide Web.HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, and Google Gears.

Browser Support

The latest versions of Apple Safari, Google Chrome, Mozilla Firefox, and Opera all support many HTML5 features and Internet Explorer 9.0 will also have support for some HTML5 functionality.The mobile web browsers that come pre-installed on iPhones, iPads, and Android phones all have excellent support for HTML5.HTML5 supports both audio and video whereas none of these was a part of HTML. HTML does not allow JavaScript to run within the web browser whereas HTML5 provides full support for JavaScript to run in the background. HTML5 supports new kinds of form controls, for example: dates and times, email, number, range, tel, URL, search etc. There are many new elements introduced in HTML5. Some of the most important ones are: summary, time, audio, details, embed, figcaption, figure, footer, header, article, canvas, nav, output, section, source, track, video, etc.

1.2.2 Bootstrap:-

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. Bootstrap 4 is the newest version of Bootstrap, which is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. Bootstrap 4 is completely free to download and use.

Bootstrap is CSS based framework developed by twitter for creating responsive web application. It provide multiple ways of implementation:-

Installation: Install Bootstrap’s source Sass and JavaScript files via npm, Composer, or Meteor.

Bootstrap Themes: Themes are built on Bootstrap as their own extended frameworks, rich with new components and plugins, documentation, and powerful build tools.

Bootstrap Icons: For the first time ever, Bootstrap has its own open source SVG icon library, designed to work best with our components and documentation.Bootstrap Icons are designed to work best with Bootstrap components, but they’ll work in any project. They’re SVGs, so they scale quickly and easily, can be implemented in several ways, and can be styled with CSS.

BootstrapCDN: It is used to build a quick responsive website provided for free by the folks at Stack Path. BootstrapCDN is a public content delivery network. Users of BootstrapCDN can load CSS, JavaScript and images remotely, from its servers using the following links.

```
<-- Bootstrap CSS -->

<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">

<!--JavaScript -->

<!—jQuery --- >
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"crossorigin="anonymo
us"></script>

< ---- Popper.js --- >

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"integr
ity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"crossorigin="anonymo
us"></script>

<--- Bootstrap JS --- >
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"integrity="s
ha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"crossorigin="anonymo
us"></script>
```

Bootstrap is built to work best in the latest desktop and mobile browsers, meaning older browsers might display differently styled, though fully functional, renderings of certain components.

Supported browsers

Specifically, we support the latest versions of the following browsers and platforms.

Alternative browsers which use the latest version of WebKit, Blink, or Gecko, whether directly or via the platform's web view API, are not explicitly supported. However, Bootstrap should (in most cases) display and function correctly in these browsers as well. More specific support information is provided below.

Mobile devices

Generally speaking, Bootstrap supports the latest versions of each major platform's default browsers. Note that proxy browsers (such as Opera Mini, Opera Mobile's Turbo mode, UC Browser Mini, Amazon Silk) are not supported.

Table 1: Browser support on mobile devices

	Chrome	Firefox	Safari
Android	Supported	Supported	N/A
iOS	Supported	Supported	Supported

Desktop browsers

Similarly, the latest versions of most desktop browsers are supported.

Table 2: Browser support for desktop devices

	Chrome	Firefox	Internet Explorer	Opera	Safari
Mac	Supported	Supported	N/A	Supported	Supported
Windows	Supported	Supported	Supported	Supported	Not supported

On Windows, we support Internet Explorer 8-11.
For Firefox, in addition to the latest normal stable release, we also support the latest Extended Support Release (ESR) version of Firefox.
Unofficially, Bootstrap should look and behave well enough in Chromium and Chrome for Linux, Firefox for Linux, and Internet Explorer 7, as well as Microsoft Edge, though they are not officially supported.

1.3 Back End

1.3.1 Python:-

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985-1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. Key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Characteristics of Python

Following are important characteristics of Python Programming −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

1.4 Libraries Used

1.4.1 Tensor Flow

TensorFlow is an open source software library for numerical computation using data-flow graphs. It was originally developed by the Google Brain Team within Google's Machine Intelligence research organization for machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well. It reached version 1.0 in February 2017, and has continued rapid development, with 21,000+ commits thus far, many from outside contributors. This article introduces TensorFlow, its open source community and ecosystem, and highlights some interesting TensorFlow open sourced models.

History: A couple of years ago, deep learning started to outperform all other machine learning algorithms when giving a massive amount of data. Google saw it could use these deep neural networks to improve its services:Gmail, Photo, Google search engine. They build a framework called **Tensorflow** to let researchers and developers work together on an AI model. Once developed and scaled, it allows lots of people to use it.

It was first made public in late 2015, while the first stable version appeared in 2017. It is open source under Apache Open Source license. You can use it, modify it and redistribute the modified version for a fee without paying anything to Google.

TensorFlow Architecture

Tensorflow architecture works in three parts:

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

It is called Tensorflow because it takes input as a multi-dimensional array, also known as **tensors**. You can construct a sort of **flowchart** of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output.This is why it is called TensorFlow because the tensor goes in it flows through a list of operations, and then it comes out the other side.

TensorFlow hardware, and software requirements can be classified into

Development Phase: This is when you train the mode. Training is usually done on your Desktop or laptop.
Run Phase or Inference Phase: Once training is done Tensorflow can be run on many different platforms. You can run it on

- Desktop running Windows, macOS or Linux
- Cloud as a web service
- Mobile devices like iOS and Android

You can train it on multiple machines then you can run it on a different machine, once you have the trained model.The model can be trained and used on GPUs as well as CPUs. GPUs were initially designed for video games. In late 2010, Stanford researchers found that GPU was also very good at matrix operations and algebra so that it makes them very fast for doing these kinds of calculations. Deep learning relies on a lot of matrix multiplication. TensorFlow is very fast at computing the matrix multiplication because it is written in C++. Although it is implemented in C++, TensorFlow can be accessed and controlled by other languages mainly, Python.

TensorFlow is the best library of all because it is built to be accessible for everyone. Tensorflow library incorporates different API to build at scale deep learning architecture like CNN or RNN. TensorFlow is based on graph computation; it allows the developer to visualize the construction of the neural network with Tensorboad. This tool is helpful to debug the program. Finally, Tensorflow is built to be deployed at scale. It runs on CPU and GPU.

Tensorflow attracts the largest popularity on GitHub compare to the other deep learning framework.

1.4.2 Keras

KERAS is an Open Source Neural Network library written in Python that runs on top of Theano or Tensorflow. It is designed to be modular, fast and easy to use. It was developed by François Chollet, a Google engineer.

Keras doesn't handle low-level computation. Instead, it uses another library to do it, called the "Backend. So Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano.

Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function. Keras doesn't handle Low-Level API such as making the computational graph, making tensors or other variables because it has been handled by the "backend" engine.

1.4.3 Numpy

NumPy is a python package, primarily used for scientific and numerical computing. NumPy is a portmanteau of two words, coined by the blending of "Numerical" and "Python". It is very famous among data scientists and analysts for its efficiency(run time speed) and a wide range of array operations, it provides. In this post, we will be getting acquainted with the NumPy library. NumPy was originally developed as "numeric" by Jim Hugunin in the late 90s. The present version of numpy was created by Travis Oliphant in 2005 by incorporating features from competing for numarray into numeric.

NumPy Library

NumPy library lets us store and work on a large amount of dense data effectively and efficiently. In NumPY data is stored in the form of arrays. These arrays are the crux of this library. Arrays in NumPy are similar to python built-in type "list". The closest approximate to a numpy array is a C array. This can be attributed to the fact that NumPy is written primarily in C language. But numpy arrays provide more efficient data storage and operations compared to both.

Characteristics

- Support n-dimensional arrays: NumPy arrays are multidimensional or n-dimensional lists or matrices of fixed size with homogeneous elements(i.e. data type of all the elements in the array is the same).
- Require a contiguous allocation of memory: This improves the efficiency as all elements of the array become directly accessible at a fixed offset from the starting of the array.
- Support selection using crop, slice, choose, etc.
- Support vectorised and complex operations.

- Support linear algebra, Fourier transform, and sophisticated broadcasting functions.
- In nutshell, NumPy provides capabilities similar to C/C++ but the simplicity of the use of Python.

Advantages of NumPy

- The core of Numpy is its arrays. One of the main advantages of using Numpy arrays is that they take less memory space and provide better runtime speed when compared with similar data structures in python(lists and tuples).
- Numpy support some specific scientific functions such as linear algebra. They help us in solving linear equations.
- Numpy support vectorised operations, like elementwise addition and multiplication, computing Kronecker product, etc. Python lists fail to support these features.
- It is a very good substitute for MATLAB, OCTAVE, etc as it provides similar functionalities and supports with faster development and less mental overhead(as python is easy to write and comprehend)
- NumPy is very good for data analysis.

Disadvantages of NumPy

- Using “nan” in Numpy: “Nan” stands for “not a number”. It was designed to address the problem of missing values. NumPy itself supports “nan” but lack of cross-platform support within Python makes it difficult for the user. That’s why we may face problems when comparing values within the Python interpreter.
- Require a contiguous allocation of memory: Insertion and deletion operations become costly as data is stored in contiguous memory locations as shifting it requires shifting.

1.4.4 Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

In short, Pandas is a Software Library in Computer Programming. It is written for the Python Programming Language. They are used in Python to deal with data analysis and manipulation. To put it in simpler words, Pandas help us to organize data and manipulate the data by putting it in a tabular form.

Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn.Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots.

This tool is essentially your data's home. Through pandas, you get acquainted with your data by cleaning, transforming, and analysing it.For example, say you want to explore a dataset stored in a CSV on your computer. Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like
- What's the average, median, max, or min of each column?
- Does column A correlate with column B?
- What does the distribution of data in column C look like?
- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria
- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.
- Store the cleaned, transformed data back into a CSV, other file or database

1.4.5 Matplotlib

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers .

Matplotlib was originally written by John D. Hunter in 2003. The current stable version is 2.2.0 released in January 2018.

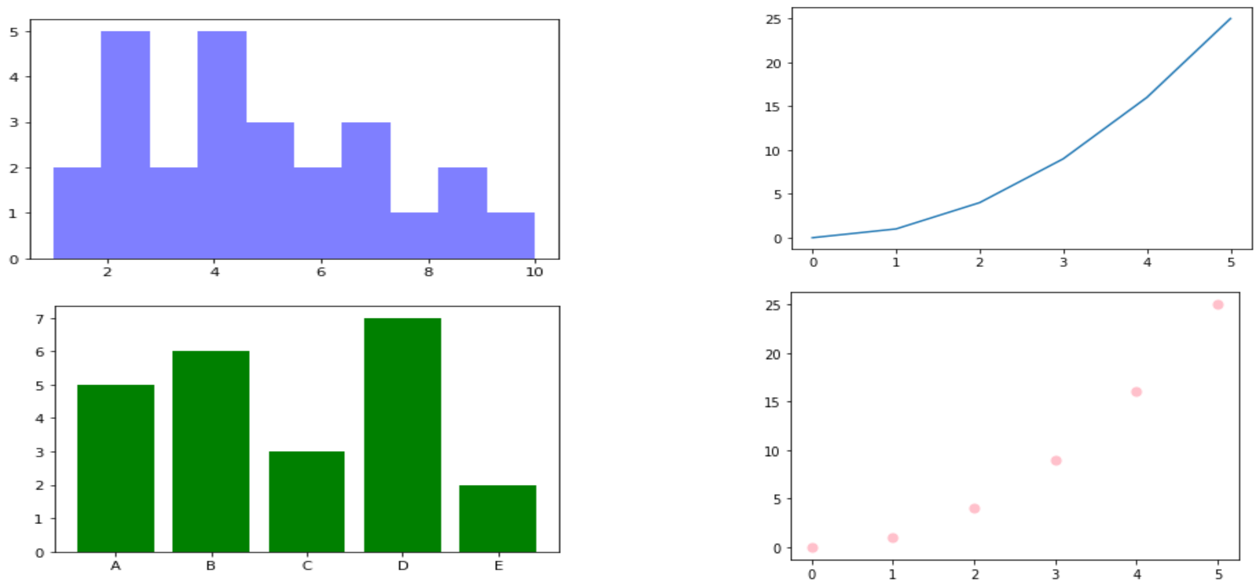


Figure 12: Types of graph

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.It tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code.

1.5 Framework Used

1.5.1 Django

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support.

History

Django was initially developed between 2003 and 2005 by a web team who were responsible for creating and maintaining newspaper websites. After creating a number of sites, the team began to factor out and reuse lots of common code and design patterns. This common code evolved into a generic web development framework, which was open-sourced as the "Django" project in July 2005.

Django has continued to grow and improve, from its first milestone release (1.0) in September 2008 through to the recently-released version 2.0 (2017). Each release has added new functionality and bug fixes, ranging from support for new types of databases, template engines, and caching, through to the addition of "generic" view functions and classes (which reduce the amount of code that developers have to write for a number of programming tasks).

Django helps you write software that is:

Complete

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and up-to-date documentation.

Versatile

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, XML, etc). Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

Secure

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash.

A password hash is a fixed-length value created by sending the password through a cryptographic hash function. Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password.Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and clickjacking (see Website security for more details of such attacks).

Scalable

Django uses a component-based “shared-nothing” architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two).

Maintainable

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the Model View Controller (MVC) pattern).

Portable

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavours of Linux, Windows, and Mac OS X. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites.

2. LITERATURE REVIEW

Face recognition and the classification of the age and gender of face objects are two interesting field of research in this area. In this paper we provide a brief review of some of existing methods in face, gender and age recognition.

In Dobry et al.(2009), anchor modeling utilizes a back end SVM to model the distribution of similarity scores between training data and all the anchor speaker models. Due to the different aspects of modeling, combining different classification methods together can often significantly improve the overall performance (M¨uller and Burkhardt, 2007; van Heerden et al., 2010; Meinedo and Trancoso,2010; Bocklet et al., 2010; Kockmann et al., 2010; Lingenfelser et al., 2010).Humans perceive gender not only based on the face, but also on the surrounding context such as hair, clothing and skin tone [15, 6], gait [14] and the whole body [6, 1]. Below, we review relevant work on gender prediction from facial images only. Theproblem of gender classification based on human faces has been extensively studied in the literature [20, 3].There are two popular methods. The first one is proposed by Moghaddam et al. [20] where a Support Vector Machine (SVM) is utilized for gender classification based on thumbnail face images. The second was presented by Baluja et al.[3] who applied the Adaboost algorithm for gender prediction. Recently, due to the popularity of Local Binary Patterns (LBP) in face recognition applications [2], Yang et al.[30] used LBP histogram features for gender feature representation, and the Adaboost algorithm to learn the best local features for classification. Experiments were performed to predict age, gender and ethnicity from face images. . Other local descriptors have also been adopted for gender classification. Wang et al. [27] proposed a novel gender recognition method using Scale Invariant Feature Transform (SIFT) descriptors and shape contexts. Once again, Adaboost was used to select features from face images and form a strong classifier. Gao et al. performed face-based gender classification on consumer images acquired from a multi-ethnic face database. To overcome the non-uniformity of pose, expression, and illumination changes, they proposed the usage of Active Shape Models (ASM) to normalize facial texture. The work concluded that the consideration of ethnic factors can help improve gender classification accuracy in a multi-ethnic environment.

2.1. Age Recognition

Not only the present studies, in the 90s used the analysis of facial geometry to estimate the age of a person, but also more recent techniques like the pipeline used in, presenting a combination of Biologically Inspired Features (BIF) and then using Canonical Correlation Analysis (CCA) andPartial Least Square (PLS) based methods. Indeed BIF were already used in to represent face images, paving the way to works like, showing that the automatic approach had matched the human performance. Infact, most of the approaches previous to CNNs were based on a two-stagepipeline, i.e. extracting features such as Local Binary Patterns (LBP),and then classifying with a Support Vector Machine (SVM), or a Multi-layer

Perceptron (MLP). On the contrary, CNN based methods typically implements the two-step pipeline described above in just one step: thenetwork learns both extracting the best features and either classifying suchfeatures into age categories or performing age regression.

Deeper CNN models have been also applied to age and gender recognition, although most of them depend on domain-specific pre-training .Cascaded combinations of deep models were also considered. CNNs for facial images analysis have not been restricted to age estimation, but also to face verification, facial attribute estimation, and genderrecognition. One illustrative example is the method which achieves a 99.2% face verification accuracy on the challenging Labelled Faces in the Wild dataset. Unfortunately, this so impressive performance hasnot been yet achieved in other facial analysis tasks like gender recognition,for example, as shown next.

2.2. Gender Recognition

Regarding gender recognition, in contrast to age analysis, there is workfrom the early 90s where neural networks were already proposed, like thepioneering approach presented. Authors proposed two neural networkstructures, an auto encoder and a classier whose input was the encodedoutput layer of the auto encoder. The drawback of this method was that itrelied on manual cropping, scaling and rotating the face of the picture, whichwas taken in a controlled environment.

Inspired from the age estimation methodology, pipelines based on a feature extractor and a stacked classier were also proposed. On the other hand, the same CNN-based methods used for agewere also applied to gender demonstrating that CNNs are trulycapable of learning how to perform different tasks without any modificationbesides the data used for learning. CNN is trained toperform gender recognition by _ne-tuning a pre-trained network, and then an SVM is trained using the deep features computed by the CNN.

Table 3: Summary of Literray review

Sno	Title of paper	Author	Publication	Technique Used
I.	Age and Gender Estimation of Unfiltered Faces	Eran Eidinger, Roee	IEEE TRANSACTIONDEC. 2014	Robust face alignment technique, SVM
II.	Automated	Yaoyu Tao	Stanford, CA 94305, USA	LBP & Gabor filter

	Estimation of Human Age, Gender and Expression		taoyaoyu@stanford.edu	LDA algorithm
III.	Comparison of Recent Machine Learning Techniques for Gender Recognition from Facial Images	Joseph Lemley Sami Abdul-Wahid Dipayan Banik	Central Washington University Ellensburg, WA, USA MAICS 2016	Feature extraction techniques: PCA & HOG. Gender classification methods
IV.	Age Group Estimation using Face Features	Ranjan Jana, Debaleena Datta, Rituparna	(IJEIT) Volume 3, Issue 2, August 2013	K-means clustering algorithm. PCA, LDA.
V.	Partial Face Recognition: Alignment-Free Approach	Shengcai Liao, Anil K. Jain, Fellow, IEEE and Stan Z. Li	IEEE transactions on pattern analysis	PCA + LDA & LBP Canny edge detecto
VI.	Gender Recognition and Age-group Prediction: A Survey	Mr. Brajesh Patel. Mr. Raghvendra	ISSN:2319-7242 Volume 3 Dec.2014	Algorithm: SVM Adaboost
VII.	Gender Recognition from Model's Face SVM Algorithm	Deepak Deshmukh	(IJETT)-Volume 10 Number 1 - Apr 2014	(IJETT)-Volume 10 Number 1 - Apr 2014
VIII.	Combining Face and Iris Biometrics for Identity Verification	Yunhong Wang, Tieniu Tan, Anil K. Jain	Center for Biometrics Authentication & testing	Algorithms:PCA, ICA ,LDA. Eigenface method as face matcher
IX.	An Image Mining System for Gender Classification & Age Prediction Based on Facial Features	Ms.Dhanashri Shirkey , Prof.Dr.S.R. Gupta	e-ISSN: 2278 Volume 10, Issue 6 (May. - Jun. 2013)	Adaboost tool for feature selection. Viola's method
X.	Face Recognition Based on Improved SIFT Algorithm	Ehsan sadeghipo Nasrollah sahragard	(IJACSA) Vol. 7, No. 1, 2016	Improved SIFT descriptor using Gabor

3. PROPOSED METHODOLOGY

In the project, the dataset images are input into the algorithm to identify gender. The dataset is used for training purpose of the model. A large amount of dataset is used for training the CNN model. Here, the CNN is used as a classifier/ algorithm. Convolutional networks were inspired by biological processes. It require less amount of pre-processing in comparison with other image classifiers. They are often used in image recognition systems.

As shown in Fig. 14, the input given is in the form of a face image to the pre-processing unit. The pre-processing unit analyses the image features based on the algorithms. Data pre-processing for machine learning is a technique that is used to convert the raw data into a clean dataset i.e. whenever the data is gathered from different sources, it is collected in raw format and raw format is not appropriate for the analysis. After pre-processing the data, the model is trained using this clean dataset. An unknown image is then inputted to predict the gender of the image.

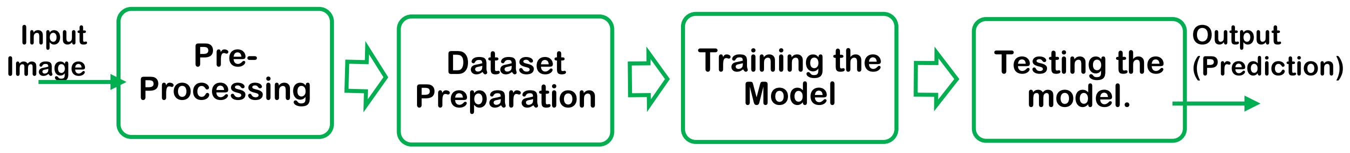


Figure 13: Basic Block Diagram

Input Image

Input image is the image intended to test with the age and gender classifier. User can input any type of image format like .jpg, .png, .tiff, and .bmp. System will not accept face images with spectacles and images of more than one people at a time.

Face Detection

First phase will proceed to check whether the given input image contains a face image or not. Algorithm will reject the input image if there isn't any face area in the input image. If facial images detected, classifier will identify the face areas from the images and create a separate image per every face in the input image as shown in Figure 12. Classifier has been trained with a sufficient number of frontal, nearly frontal, rotated faces from 0 to 45 degrees and non-face images. Detected face images are pre-processed to standardize the face images by converting them to a unique format.

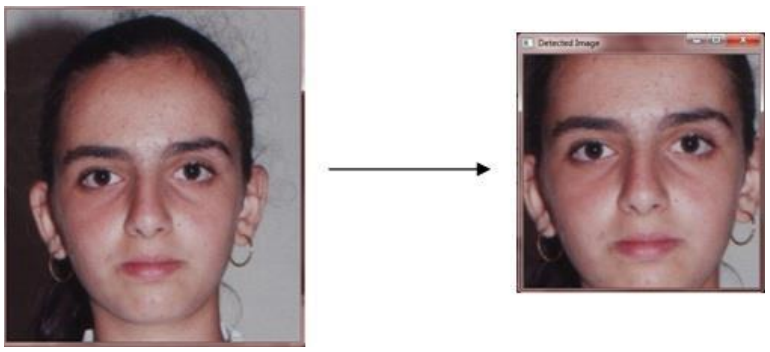


Figure 14: Detected face area from input image

Pre-processing

Images used in the experiment are in different conditions such as presence of noisy data, different lighting conditions and different intensity levels. Thus, detected face images need to undergo a pre-processing step before forwarding to the classification stage.

1. Resize Detected Face Image -Collected images from the initial face detection are in different sizes. Therefore to standardize the data set first step in preprocessing is to modify each image into a standard width and height (Ex. 255*255 in this research).

2. Colour Conversion -Images used in this research need to be in standard colour format. Therefore to overcome the complexity, all the images are converted into grayscale and finally do a histogram equalization to have a uniform distribution of intensity values in the image. First the red, green and blue values of every pixel in the image are obtained. The following function is used to convert RGB into

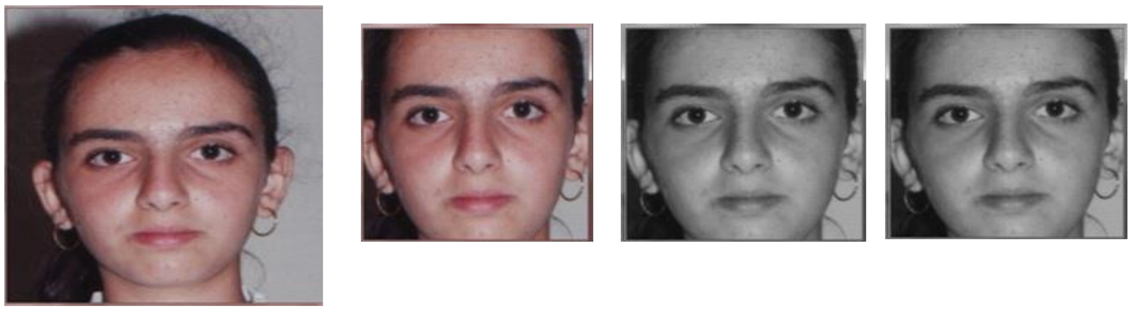


Figure 15: (a) Input image (b) Face detected (c) greyscale image (d) Smoothed

Greyscale image.

As shown in Fig. 16, the model consists of five stack of layers. The layers include Convolutional layer, Activation layer, Max Pooling layer, Flatten layer, and dense layer and Dropout layer. The first three stacks consist of Convolutional layer, Activation layer and Max Pooling layer. The activation layer used is Rectified Linear Unit (ReLu). The fourth stack consists of Flatten layer, dense layer and Activation layer. After the first three layers, the output is in 3D and needs to be converted to 1D form. The Flatten layer is used to convert the 3D output to 1D format.The Dense layer is also known as Fully Connected layer. It is used to convert the matrix into a list format and all the nodes are connected to each other. The Activation layer used is Rectified Linear Unit (ReLu).

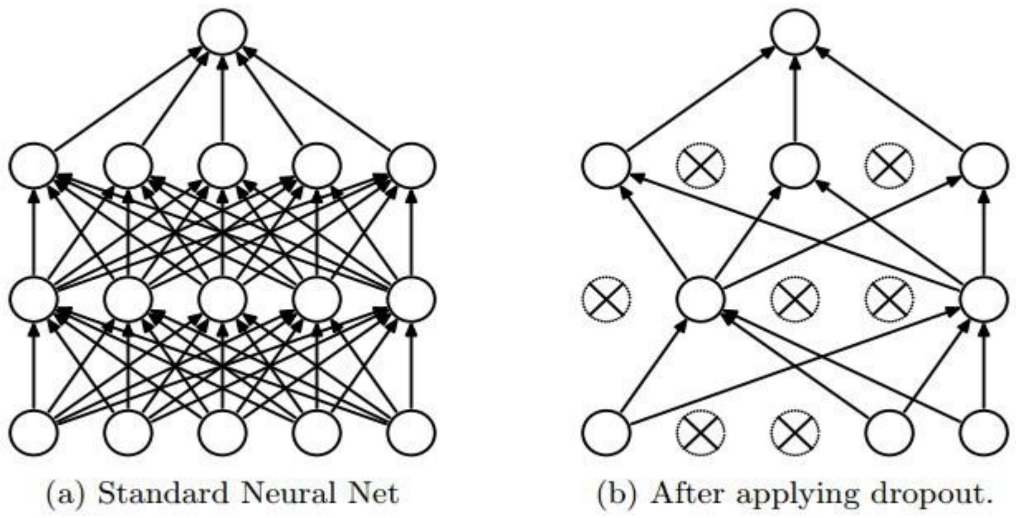


Figure 16: Dropout Layer

The fifth stack consists of Dropout layer, dense layer and Activation layer. A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which curbs the individual power of each neuron leading to over-fitting of training data. The Dropout layer is used to remove the duplicate images present in the dataset to avoid system to undergo overfitting. The Activation layer used is Sigmoid. The sigmoid classification is best for binary classification problems such as gender. Generally, the network is trained using a larger and domain related dataset. After the convergence of the network parameters, an extra training step is done to optimize the network weights using in-domain data.

3.1 Brief Explanation

To start with the project, the first step that needs to be done is data collection. Datasets play an important in deep learning as it is used to train the system to get the required output. Some datasets are available publicly while some are not. The UTKFace is a publicly available dataset which can be used for gender and age classification. This dataset has images of people of different age, gender and age. In the project, the dataset images are input into the algorithm to identify the gender.

The UTKFace dataset is used to train the model to perform gender and age classification. The input given is in the form of a face image and the image features are analysed based on the algorithm. An unknown image is then inputted to predict the gender and age of the same. An output will be generated which will contain the prediction of the unknown face image.

The UTKFace dataset is categorized into 'Train' and 'Validation', each of which contains 'Male' and 'Female'. The project is trained using 8,000 images in each class and validated using 1,000 images in each class. The model is trained using ConvNet (Convolutional Neural Network) consisting of 2 layers. The CNN consists of many hidden layers such as Convolutional layer, ReLu layer, Max Pooling layer, Fully Connected layer, etc. Using these layers, the input face image is converted into weights and saved in '.h5' format. These weights are then used to predict an unknown image. The average accuracy achieved in the project is 90%.

The training program includes data augmentation. Data augmentation means increasing the number of images in the dataset because plentiful high-quality information is the key to significant machine learning models. Foremost, training examples needs to be augmented via a variety of random transformations, so that the model would never see twice the exact same picture and this helps prevention of overfitting thereby generalizing model in a better way.

In the project, Keras is used to work on Tensorflow. Keras is an open source neural network library. It is user-friendly and provides many features like activation function, layers, optimizers, etc. Keras support both CNN and RNN. Using Java Virtual Machine (JVM), deep models can be created on iOS and Android. This can be achieved by using an appropriate class in keras. Convnet is the right tool for image classification. Data augmentation is a way to fight overfitting, however, it is not enough since the augmented samples are still highly correlated. The main focus for fighting overfitting must be the entropic capability i.e. the abundant information that the model is allowed to store. A model which can store a lot of information has the potential to be more accurate by leveraging more features in comparison to a model that can only store a few features. But the former is also more at risk as it will start storing irrelevant features whereas a model that can only store a few features will have to focus on the most significant features found within the data. The one which stores fewer features are more likely to be truly relevant and are easy to generalize. The setup for the project is as follows:

- 224,250 training examples (8000 per class)
- 5750testing examples (1000 per class)

TrainingDataset: The training dataset is a set of examples employed to train the model i.e. to fit the parameters. Most of the approaches used for training the samples tend to over fit if the dataset is not increased and used in variety.

Test Dataset: The test dataset is not dependent on the training or validation dataset. If a model is fitting both the training dataset as well as test dataset then it can be said that minimum overfitting has taken place. The test dataset is employed to check the performance characteristics like the accuracy, loss, sensitivity, etc.

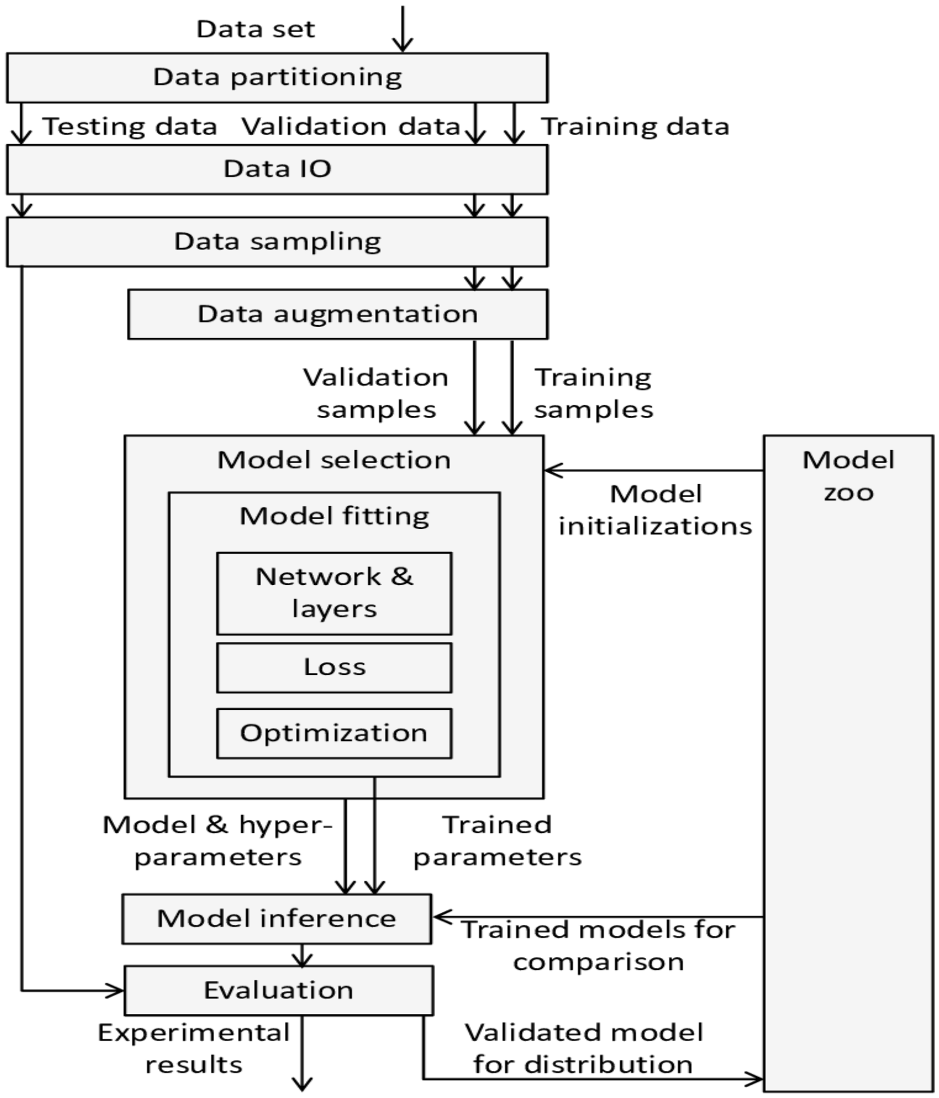


Figure 17: Processing

As shown in fig.19the dataset is partitioned into two parts training and testing dataset. Training dataset consists of 75% of the whole dataset and the testing data comprises of 25%. Data sampling is done. Data sampling is a statistical analysis technique used to select, manipulate and analyse a representative subset of data points to identify patterns and trends in the larger data set being examine. There are two ways of sampling data as shown below.

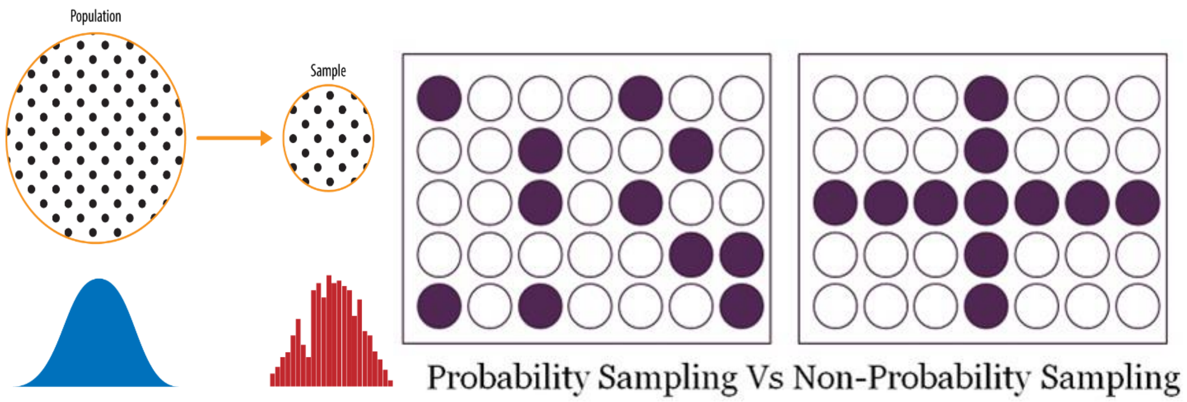


Figure 18: Sampling Of Data

Image augmentation is a very powerful technique used to artificially create variations in existing images to expand an existing image data set. This creates

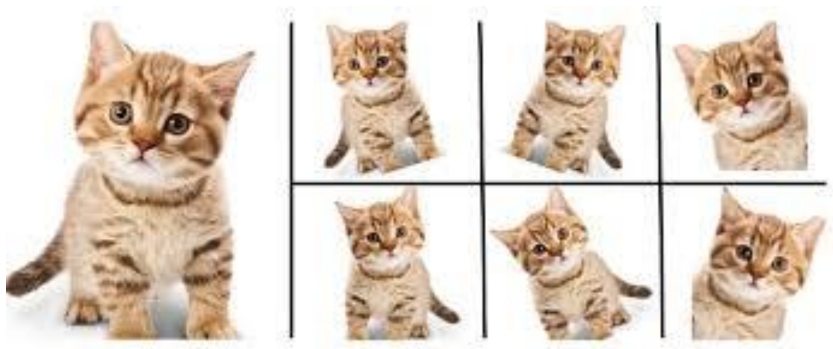


Figure 19: Augmentation of image by Keras

new and different images from the existing image data set that represents a comprehensive set of possible images. This helps to increase the performance of the model by generalizing better and thereby reducing overfitting.A CNN, due to its in-variance property, can classify objects even when visible in different sizes, orientations or different illumination.

Keras permits the model to perform random transformations and normalization operations on batches of image data. The attributes used are rotation range, width shift and height shift, rescale, shear range, zoom range, horizontal rip and fill mode. By using these attributes, the system can automatically rotate pictures, translate pictures, rescale pictures, zoom into pictures, apply shearing transformations, rip images horizontally, fill newly created pixels, etc.

After this the data is inputted in the model and passes through different layers of neural network and gets trained Machine learn by means of a loss function. It’s a method of evaluating how well specific algorithm models the given data. If predictions deviates too much from actual results, loss functionwould cough up a very large number. Broadly, loss functions can be classified into two major categories depending upon the type of learning task we are dealing with Regression losses and Classification losses.

Deep learning is a highly iterative process. We have to try out various permutations of the hyper parameters to figure out which combination works best. Therefore it is crucial that our deep learning model trains in a shorter time without penalizing the cost. Gradually, with the help of some optimization function, loss function learns to reduce the error in prediction. The performance of the optimization algorithm directly affects the model's training efficiency.

The model is tested with validation dataset through the process and a visual representation is used to show the progress in learning process of the model.

3.1.1 Methodology Used

The methodology used is Iterative waterfall model. In this model, the software development activities move to the next phase only after the activities in the current phase are over. However, unlike in the case with a waterfall model, one cannot return to the previous stage in case of Iterative Waterfall Model.

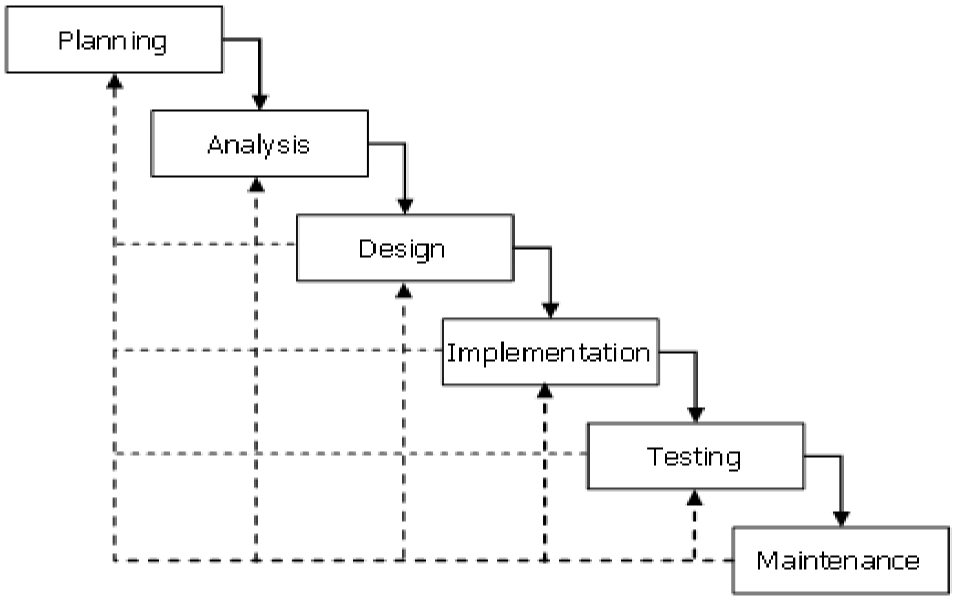


Figure 20: Diagram showing Water Fall Model

The most important advantage of the iterative waterfall model lies in the fact, that there is minimum planning overhead for the steps that are to follow, since the activities in each of the phase is carried out upfront, it is feasible that one does not have to plan for the entire phase. The most important of the advantage is that the project does not slip on its schedule. The number of resources working on the project does not keep on increasing with each passing day, as the planning for the same is done at the start of the phase itself.

3.2 Graphs and Diagrams

3.2.1 Data Visualization

Visualization of the performance of any machine learning model is an easy way to make sense of the data being poured out of the model and make an informed decision about the changes that need to be made on the parameters or hyperparameters that affects the Machine Learning model.

Reasons to Visualize

- To evaluate the Under fitting or Overfitting: Visualizing the training loss vs. validation loss or training accuracy vs. validation accuracy over a number of epochs is a good way to determine if the model has been sufficiently trained. This is important so that the model is not undertrained and not over trained such that it starts memorizing the training data which will, in turn, reduce its ability to predict accurately.
- To adjust the Hyperparameters: Hyperparameters such as the number of nodes per layer of the Neural Network and the number of layers in the Network can make a significant impact on the performance of the Model. Visualization of the fitness of the training and validation set data can help to optimize these values and in building a better model.

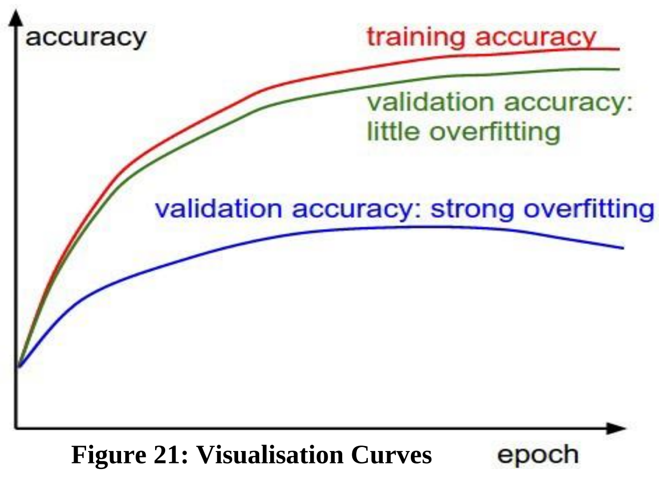


Figure 21: Visualisation Curves

3.2.1.1 Loss Curve

One of the most used plots to debug a neural network is a Loss curve during training. It gives us a snapshot of the training process and the direction in which the network learns. A loss function is used to optimize a machine learning algorithm. The loss is calculated on training and validation and its interpretation is based on how well the model is doing in these two sets. It is the sum of errors made for each example in training or validation sets. Loss value implies how poorly or well a model behaves after each iteration of optimization.

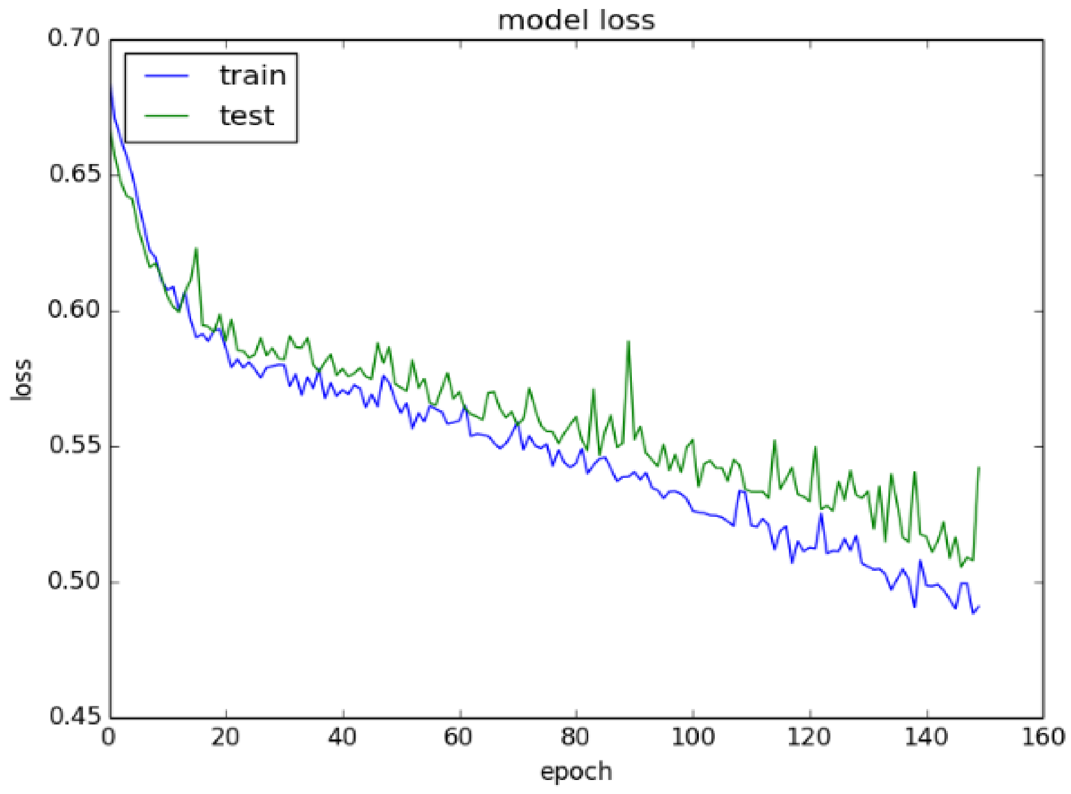


Figure 22: Loss Curve

3.2.1.2 Accuracy Graph

After building a model on a training set, to ensure that the model is not an overfit for the data, the test set was used to evaluate the model accuracy. An accuracy metric is used to measure the algorithm’s performance in an interpretable way. The accuracy of a model is usually determined after the model parameters and is calculated in the form of a percentage. It is the measure of how accurate your model's prediction is compared to the true data.

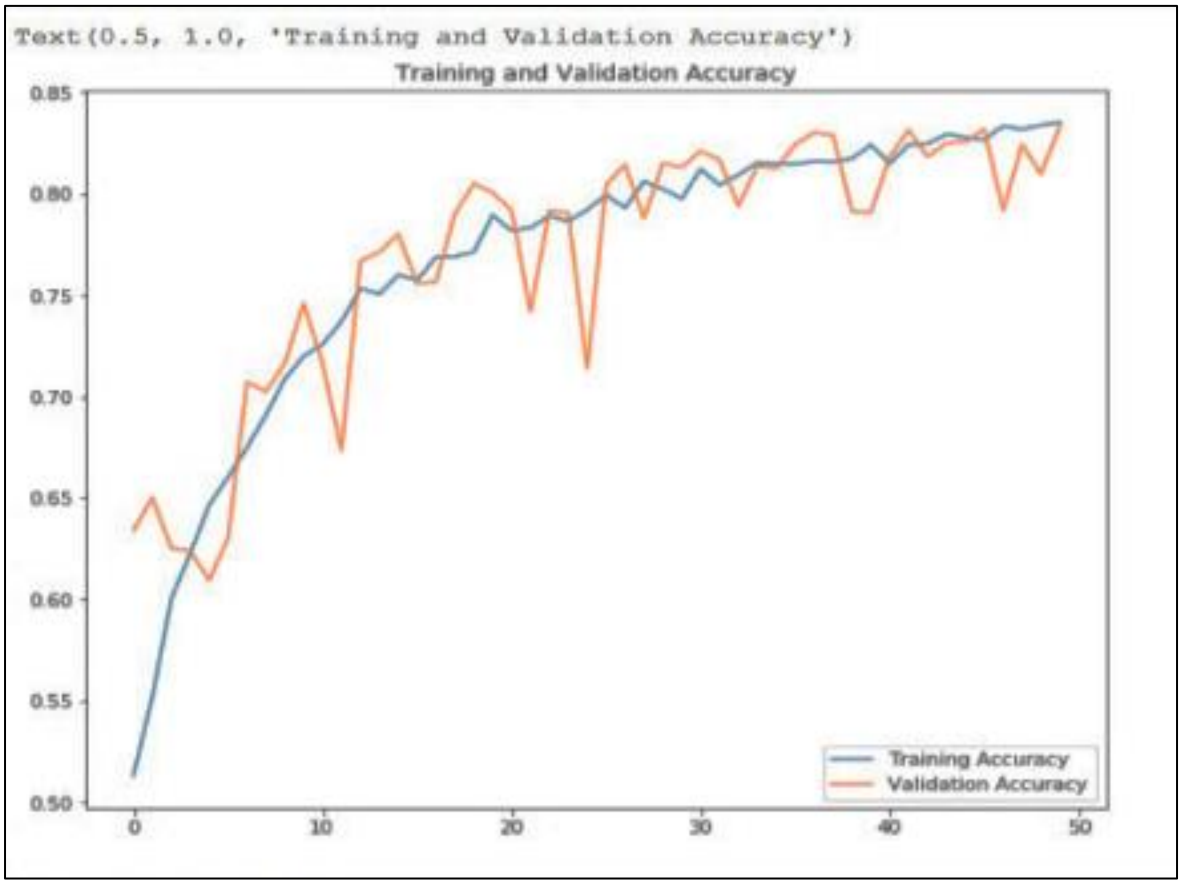

```
Text(0.5, 1.0, 'Training and Validation Accuracy')
```

Figure 23: Training and validation accuracy graph

3.2.2 E R Diagram

ENTITY-RELATIONSHIP DIAGRAM (ERD) displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.Here, are prime reasons for using the ER Diagram:-

- Provide a preview of how all your tables should connect, what fields are going to be on each table.
- Helps to describe entities, attributes, relationships.
- ER diagrams are translatable into relational tables which allows you to build databases quickly.
- ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications.
- The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram.
- ERD is allowed you to communicate with the logical structure of the database to users.

As mentioned earlier the applications contain a subscription page, the page contains a number of entries which needs to be filed by the user. E R diagram in Fig 20 shows details of the subscription page.

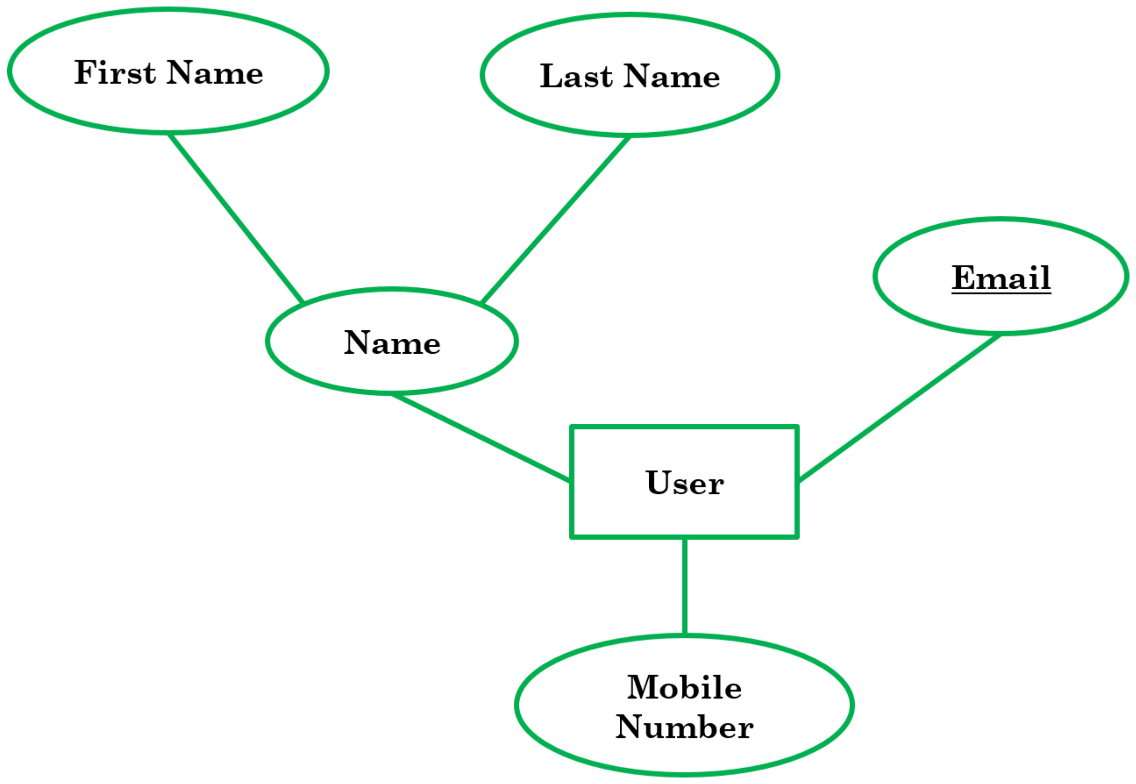


Figure 24: E R Diagram

ER- Diagram is a visual representation of data that describe how data is related to each other. The rectangle represents the entity type which is user in this case. Text enclosed in ellipse symbol represent attributes, name, mobile number, Email are various attributes. Here name is a multivalued attribute consisting first and last name. Underlined attributes are known as Primary Key. In the relational model of databases, a primary key is a specific choice of a minimal set of attributes that uniquely specify a tuple in a relation. Informally, a primary key is "which attributes identify a record", and in simple cases are simply a single attribute: a unique id. Lines are used to link the attributes to entity type and to other relation.

3.2.3 Data Flow Diagram

A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. The following diagram shows the division and flow of dataset used in this project.

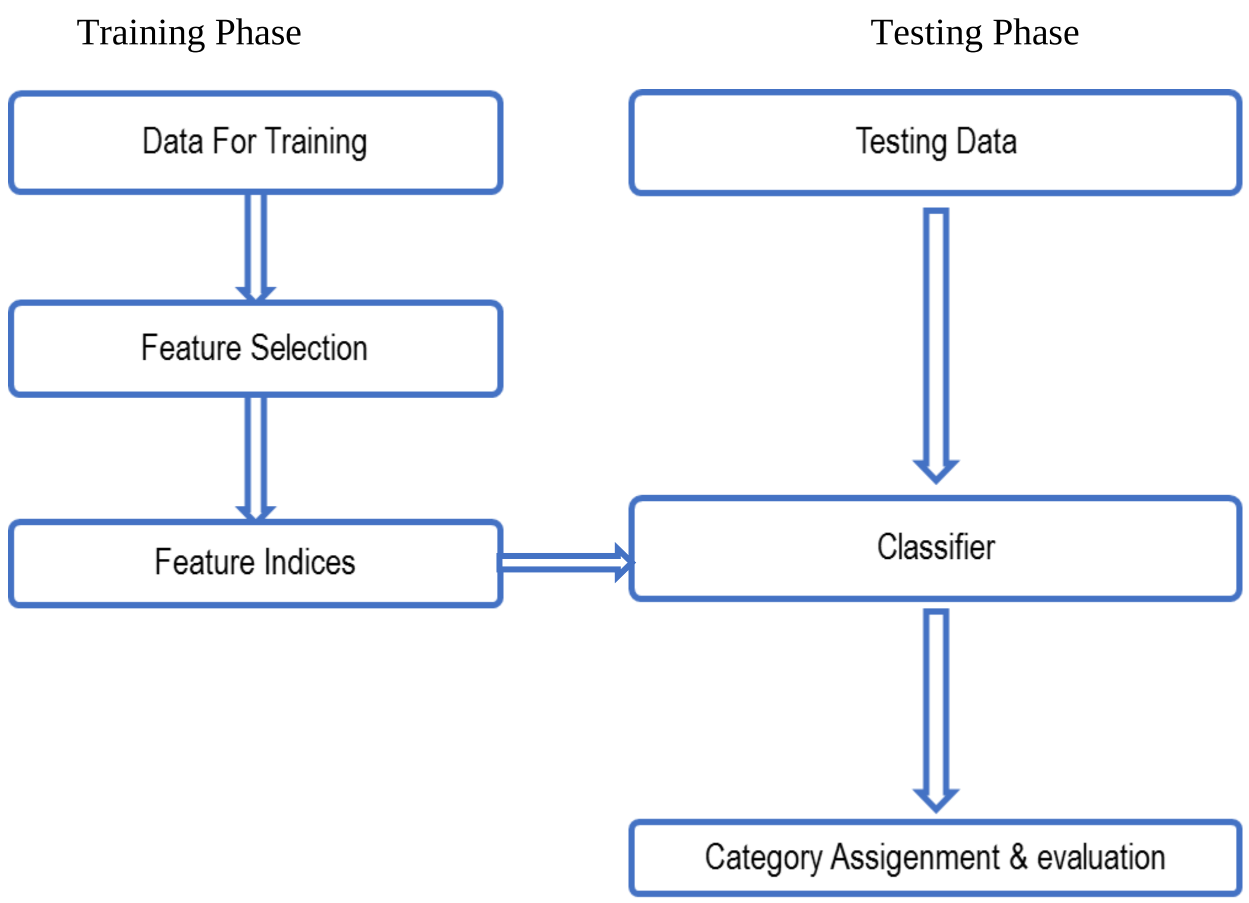


Figure 25: Diagram representing DFD of Model

3.3 Requirments Specification

Requirement Analysis is the first technical step in the software engineering process. It is at this point that a general statement of software scope is refined into a concrete specification that becomes a foundation for all the software engineering activities that follow.Analysis must focus on the informational, functional, and behavioural domains of a problem. To better understand what is required, models are created, the problem is partitioned, and representation that depict the essence of requirements and later, implementation detail, are developed.

3.3.1 Hardware Requirement:

- **Processor** : fourth generation intel core i3
- **Speed** : 1.7 GHz.
- **RAM** : 4 GB (minimum) OR more
- **Hard Disk** : 20 GB
- **Key Board** : Standard Windows Keyboard
- **Mobile** : Mobile with Android Operating System

3.3.2 Software Requirement:

- **Operating System** : Any
- **Database** :
- **IDE** :
- **Network** : Internet with minimum speed 64kbp pr sec.

4. IMPLEMENTATION

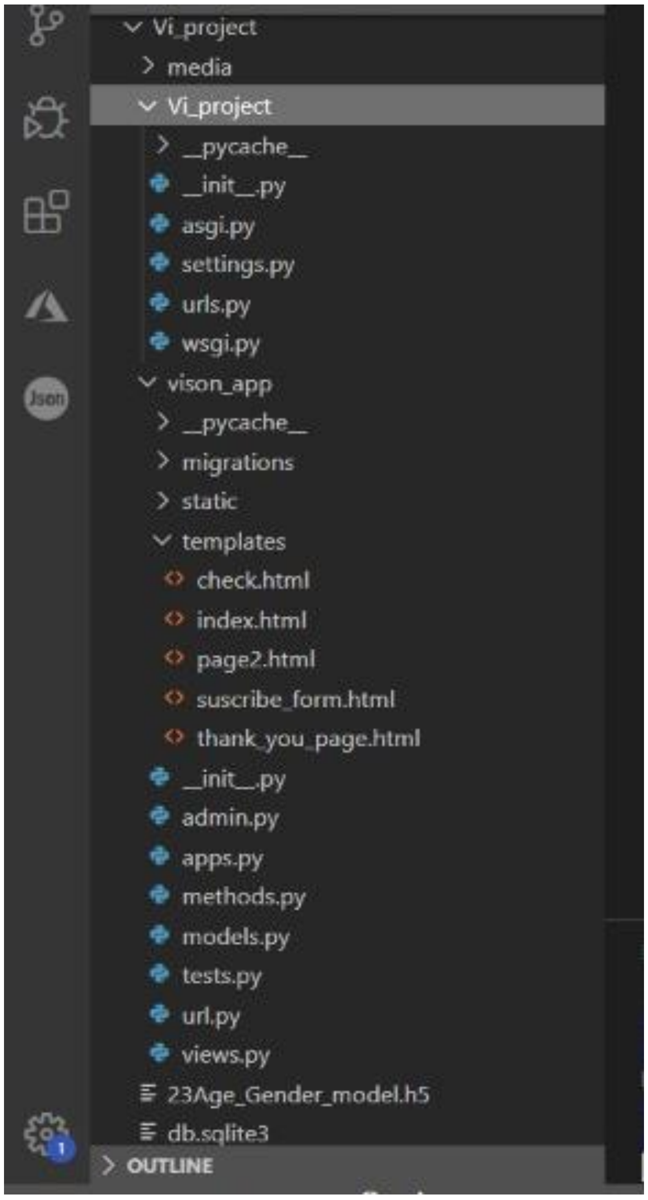

```
Vi_project
  media
  Vi_project
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
  vison_app
    __pycache__
    migrations
    static
    templates
      check.html
      index.html
      page2.html
      suscribe_form.html
      thank_you_page.html
    __init__.py
    admin.py
    apps.py
    methods.py
    models.py
    tests.py
    url.py
    views.py
  23Age_Gender_model.h5
  db.sqlite3
```

Figure 26: Model Files

4.1 Module Description

4.1.1 View.py

```
from django.shortcuts import render
from django.http import HttpResponse
from django.http import HttpResponseRedirect
from django.views.decorators.csrf import csrf_exempt
from django.conf import settings
from django.conf.urls.static import static
from django.core.files.storage import FileSystemStorage
import pandas as pd
import numpy as np
import os
import cv2
import keras
import keras.backend.tensorflow_backend as tb
from vison_app import methods
from .models import Suscribe

@csrf_exempt
def homePage(request):
return render(request,'index.html')

@csrf_exempt
def suscribe_form(request):
return render(request,'suscribe_form.html')

@csrf_exempt
def suscribe(request):
name = request.POST["name"]
last_name = request.POST["last_name"]
email = request.POST["email"]
data = Suscribe(name = name, middleName = last_name, email =
email)
data.save()
return render(request,'thank_you_page.html')

@csrf_exempt
def pic_upload(request):
try:
file=request.FILES['myfile']
fs = FileSystemStorage()
filename = fs.save(file.name, file)
upload_url=fs.url(filename)
path = "Vi_project/media/"+file.name
```

```
output= methods.read_photo(path)
if output != "No_face" and output != "Multiple_face":
try:
gender,age = methods.predition_age_gender(output)
return
render(request,'page2.html',{'pic_name':upload_url,'age':age,'ge
nder':gender})
except Exception as error:
return
render(request,'page2.html',{'error':str(error),'pic_name':uploa
d_url})
else:
return
render(request,'page2.html',{'error':output,'pic_name':upload_ur
l})
except:
return HttpResponseRedirect("/")
```

4.1.2 Model.py

Contains the database schema

```
from django.db import models


class Suscribe(models.Model):
    name = models.CharField(max_length=200)
    middleName = models.CharField(max_length=100)
    email = models.EmailField()


    def _str_(self):
        return self.name

class suscribe_in(models.Model):
    suscribe = models.CharField(max_length=200)

    def _str_(self):
        return self.suscribe
```

4.1.3 Methods.py

Contains all functions and provide reusability of code for the model

```
import json
import os
import cv2
import numpy as np
import pandas as pd
import tensorflow as tf
from keras import backend as K
from keras.models import load_model

# Create your models here.

def load_model_from_path():
graph = tf.get_default_graph()
model = load_model('Vi_project/23Age_Gender_model.h5')
return graph,model

def read_photo(path):
imm = cv2.imread(path,cv2.IMREAD_COLOR)
if imm.any():
gray = cv2.cvtColor(imm, cv2.COLOR_RGB2GRAY)
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
faces = faceCascade.detectMultiScale(
gray,
scaleFactor=1.3,
minNeighbors=3,
minSize=(60, 60)
)
if len(faces)==1:
for (x, y, w, h) in faces:
cv2.rectangle(imm, (x, y), (x + w, y + h), (0, 255, 0), 2)
roi_color = imm[y:y + h, x:x + w]
return roi_color
elif len(faces)==0:
return("No_face")
else:
return("Multiple_face")
return None

def predition_age_gender(X):
```

```
graph,model = load_model_from_path()
X_data =[]
face = cv2.cvtColor(X, cv2.COLOR_BGR2RGB)
face =cv2.resize(face, (32,32) )
X_data.append(face)
X_data=np.array(X_data)
try:
with graph.as_default():
predictions = model.predict(X_data)
gender = predictions[0]
gender = "Female" if gender>0.5 else "Male"
age = int(predictions[1])
return gender,age
except Exception as err:
raise(err)
return None
```

4.1.4 Url.py

Also known as local url file which contains all urls

```
from django.conf.urls import include
from django.urls import path

from .import views
urlpatterns = [

    path('submit/',views.suscribe,name='suscribe'),
]
```

4.1.5 Urls.py

```
"""Vi_project URL Configuration
The `urlpatterns` list routes URLs to views. For more
information please see:
https://docs.djangoproject.com/en/3.0/topics/http/urls/
Examples:
Function views
1. Add an import:  from my_app import views
2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
1. Add an import:  from other_app.views import Home
2. Add a URL to urlpatterns:  path('', Home.as_view(),
name='home')
```

```
"""
Including another URLconf
1. Import the include() function: from django.urls import
include, path
2. Add a URL to urlpatterns:  path('blog/',
include('blog.urls'))
"""

#_SYMBOLIC_SCOPE.value = True
from django.contrib import admin
from django.urls import path
from vison_app import views
from django.conf import settings
from django.conf.urls.static import static
from django.conf.urls import include, url

urlpatterns = [
path('admin/', admin.site.urls),
path('',views.homePage),
path('pic_upload/',views.pic_upload),
path('suscribe/',views.suscribe_form),
path('suscribe/',include('vison_app.url'))
]
if settings.DEBUG:
urlpatterns += static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT)
```

4.2 Training

4.2.1 Generation of model

```
def gen_model():

    inputs = keras.layers.Input(shape=(32, 32, 3))
    x = inputs
    x = layers.Conv2D(32,3,activation='relu')(x)
    x = layers.Conv2D(32,3,activation='relu')(x)
    x = layers.MaxPool2D(2)(x)
    x = layers.Dropout(0.3)(x)
    x = layers.Conv2D(64,3,activation='relu')(x)
    x = layers.Conv2D(64,3,activation='relu')(x)
    x = layers.MaxPool2D(2)(x)
    x = layers.Dropout(0.3)(x)
    x = layers.Conv2D(84,3,activation='relu')(x)
    x = layers.Dropout(0.3)(x)
    x = layers.Flatten()(x)
    x1 = layers.Dense(64,activation='relu')(x)
    x2 = layers.Dense(64,activation='relu')(x)
    x1 = layers.Dense(1,activation='sigmoid',name='sex_out')(x1)
    x2 = layers.Dense(1,activation='relu',name='age_out')(x2)
    model = keras.models.Model(inputs=inputs, outputs=[x1, x2])
    model.compile(optimizer='Adam',
loss=['binary_crossentropy','mae'])
    return model
model=gen_model()
```

If model fails or stops while training it is made to restart from the point it stopped.

```
import random
random_id=random.random()
model.summary()
callbacks = [
    keras.callbacks.EarlyStopping(patience=75,
monitor='val_loss',restore_best_weights=True),

keras.callbacks.TensorBoard(log_dir='./logs/'+str(random_id))
]
```

4.2.2 Partitioning of dataset

```
X_train, X_valid, y_train, y_valid = train_test_split(X, y,
test_size=0.33)
y_train=[y_train[:,1],y_train[:,0]]
y_valid=[y_valid[:,1],y_valid[:,0]]
```

4.2.3 Training the model

```
model.fit(X_train, y_train,
epochs=2000,batch_size=240,validation_data=(X_valid,y_valid),cal
lbacks=callbacks, shuffle=True)
```

```
Model: "model_2"
__________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
=
input_6 (InputLayer)            (None, 32, 32, 3)    0
__________________________________
conv2d_58 (Conv2D)              (None, 30, 30, 32)   896         input_6[0][0]
__________________________________
conv2d_59 (Conv2D)              (None, 28, 28, 32)   9248        conv2d_58[0][0]
__________________________________
max_pooling2d_55 (MaxPooling2D) (None, 14, 14, 32)   0           conv2d_59[0][0]
__________________________________
dropout_4 (Dropout)             (None, 14, 14, 32)   0           max_pooling2d_55[0][0]
__________________________________
conv2d_60 (Conv2D)              (None, 12, 12, 64)   18496       dropout_4[0][0]
__________________________________
conv2d_61 (Conv2D)              (None, 10, 10, 64)   36928       conv2d_60[0][0]
__________________________________
max_pooling2d_56 (MaxPooling2D) (None, 5, 5, 64)     0           conv2d_61[0][0]
__________________________________
dropout_5 (Dropout)             (None, 5, 5, 64)     0           max_pooling2d_56[0][0]
__________________________________
conv2d_62 (Conv2D)              (None, 3, 3, 84)     48468       dropout_5[0][0]
__________________________________
dropout_6 (Dropout)             (None, 3, 3, 84)     0           conv2d_62[0][0]
__________________________________
flatten_15 (Flatten)            (None, 756)          0           dropout_6[0][0]
__________________________________
dense_29 (Dense)                (None, 64)           48448       flatten_15[0][0]
__________________________________
dense_30 (Dense)                (None, 64)           48448       flatten_15[0][0]
__________________________________
sex_out (Dense)                 (None, 1)            65          dense_29[0][0]
__________________________________
age_out (Dense)                 (None, 1)            65          dense_30[0][0]
==================================================================================================
=
Total params: 211,062
Trainable params: 211,062
Non-trainable params: 0
```

```
Epoch 1/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 28.5326 - sex_out_loss: 3.4093 - age_out_loss: 25.1233 - val_loss: 24.5617 - val_sex_out_loss: 0.6668 - val_age_out_loss:
23.8949
Epoch 2/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 19.3499 - sex_out_loss: 1.0848 - age_out_loss: 18.2651 - val_loss: 26.3602 - val_sex_out_loss: 0.6687 - val_age_out_loss:
25.6914
Epoch 3/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 18.1767 - sex_out_loss: 0.7790 - age_out_loss: 17.3977 - val_loss: 25.3633 - val_sex_out_loss: 0.6731 - val_age_out_loss:
24.6902
Epoch 4/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 17.7251 - sex_out_loss: 0.7042 - age_out_loss: 17.0210 - val_loss: 22.0025 - val_sex_out_loss: 0.6496 - val_age_out_loss:
21.3529
Epoch 5/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 17.1362 - sex_out_loss: 0.6677 - age_out_loss: 16.4685 - val_loss: 23.5965 - val_sex_out_loss: 0.6253 - val_age_out_loss:
22.9712
Epoch 6/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 16.1709 - sex_out_loss: 0.6314 - age_out_loss: 15.5395 - val_loss: 21.1332 - val_sex_out_loss: 0.5838 - val_age_out_loss:
20.5494
Epoch 7/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 15.3317 - sex_out_loss: 0.5970 - age_out_loss: 14.7348 - val_loss: 16.1989 - val_sex_out_loss: 0.5401 - val_age_out_loss:
15.6588
Epoch 8/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 14.8960 - sex_out_loss: 0.5787 - age_out_loss: 14.3174 - val_loss: 15.7071 - val_sex_out_loss: 0.5405 - val_age_out_loss:
15.1666
Epoch 9/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 13.7211 - sex_out_loss: 0.5814 - age_out_loss: 13.1398 - val_loss: 15.6042 - val_sex_out_loss: 0.5624 - val_age_out_loss:
15.0418
Epoch 10/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 13.6535 - sex_out_loss: 0.5640 - age_out_loss: 13.0894 - val_loss: 14.5783 - val_sex_out_loss: 0.5268 - val_age_out_loss:
14.0515
Epoch 11/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 13.0315 - sex_out_loss: 0.5599 - age_out_loss: 12.4716 - val_loss: 15.0227 - val_sex_out_loss: 0.5174 - val_age_out_loss:
14.5053
Epoch 12/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 12.6417 - sex_out_loss: 0.5521 - age_out_loss: 12.0896 - val_loss: 15.3627 - val_sex_out_loss: 0.5218 - val_age_out_loss:
14.8409
Epoch 13/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 12.2396 - sex_out_loss: 0.5348 - age_out_loss: 11.7048 - val_loss: 12.2651 - val_sex_out_loss: 0.4901 - val_age_out_loss:
11.7749
Epoch 14/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 12.0046 - sex_out_loss: 0.5292 - age_out_loss: 11.4754 - val_loss: 14.9642 - val_sex_out_loss: 0.4930 - val_age_out_loss:
14.4712
Epoch 15/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 11.4665 - sex_out_loss: 0.5222 - age_out_loss: 10.9442 - val_loss: 15.8622 - val_sex_out_loss: 0.4874 - val_age_out_loss:
15.3748
Epoch 16/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 11.9636 - sex_out_loss: 0.5109 - age_out_loss: 11.4526 - val_loss: 12.4321 - val_sex_out_loss: 0.4794 - val_age_out_loss:
11.9527
Epoch 17/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 11.3299 - sex_out_loss: 0.4982 - age_out_loss: 10.8317 - val_loss: 12.5402 - val_sex_out_loss: 0.4568 - val_age_out_loss:
12.0834
Epoch 18/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 11.3257 - sex_out_loss: 0.4996 - age_out_loss: 10.8262 - val_loss: 13.7011 - val_sex_out_loss: 0.4478 - val_age_out_loss:
13.2533
Epoch 19/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 11.1191 - sex_out_loss: 0.4895 - age_out_loss: 10.6296 - val_loss: 14.2399 - val_sex_out_loss: 0.4465 - val_age_out_loss:
13.7934
Epoch 20/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.8082 - sex_out_loss: 0.4844 - age_out_loss: 10.3237 - val_loss: 14.7130 - val_sex_out_loss: 0.4350 - val_age_out_loss:
14.2780
Epoch 21/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.7476 - sex_out_loss: 0.4751 - age_out_loss: 10.2725 - val_loss: 12.7036 - val_sex_out_loss: 0.4258 - val_age_out_loss:
12.2778
Epoch 22/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.4718 - sex_out_loss: 0.4665 - age_out_loss: 10.0053 - val_loss: 11.3790 - val_sex_out_loss: 0.4203 - val_age_out_loss:
10.9587
Epoch 23/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.5679 - sex_out_loss: 0.4634 - age_out_loss: 10.1045 - val_loss: 11.0901 - val_sex_out_loss: 0.4082 - val_age_out_loss:
10.6819
Epoch 24/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.6992 - sex_out_loss: 0.4699 - age_out_loss: 10.2292 - val_loss: 11.9739 - val_sex_out_loss: 0.4206 - val_age_out_loss:
11.5533
Epoch 25/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.2402 - sex_out_loss: 0.4607 - age_out_loss: 9.7795 - val_loss: 15.3458 - val_sex_out_loss: 0.4395 - val_age_out_loss: 14.9063
Epoch 26/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 10.4916 - sex_out_loss: 0.4634 - age_out_loss: 10.0282 - val_loss: 13.1395 - val_sex_out_loss: 0.4026 - val_age_out_loss:
12.7368
Epoch 27/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.9566 - sex_out_loss: 0.4413 - age_out_loss: 9.5153 - val_loss: 11.9963 - val_sex_out_loss: 0.4072 - val_age_out_loss: 11.5891
Epoch 28/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.2284 - sex_out_loss: 0.4483 - age_out_loss: 9.7801 - val_loss: 15.5273 - val_sex_out_loss: 0.4026 - val_age_out_loss: 15.1247
Epoch 29/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 10.1316 - sex_out_loss: 0.4332 - age_out_loss: 9.6984 - val_loss: 12.1151 - val_sex_out_loss: 0.3883 - val_age_out_loss: 11.7269
Epoch 30/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.7303 - sex_out_loss: 0.4416 - age_out_loss: 9.2887 - val_loss: 12.9564 - val_sex_out_loss: 0.3864 - val_age_out_loss: 12.5700
Epoch 31/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.6458 - sex_out_loss: 0.4428 - age_out_loss: 9.2030 - val_loss: 14.4288 - val_sex_out_loss: 0.4042 - val_age_out_loss: 14.0246
Epoch 32/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.5919 - sex_out_loss: 0.4387 - age_out_loss: 9.1532 - val_loss: 13.6327 - val_sex_out_loss: 0.4014 - val_age_out_loss: 13.2313
Epoch 33/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4412 - sex_out_loss: 0.4304 - age_out_loss: 9.0109 - val_loss: 13.4573 - val_sex_out_loss: 0.3968 - val_age_out_loss: 13.0605
Epoch 34/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.5946 - sex_out_loss: 0.4202 - age_out_loss: 9.1744 - val_loss: 10.1899 - val_sex_out_loss: 0.3851 - val_age_out_loss: 9.8048
Epoch 35/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4297 - sex_out_loss: 0.4171 - age_out_loss: 9.0126 - val_loss: 13.1912 - val_sex_out_loss: 0.3740 - val_age_out_loss: 12.8172
Epoch 36/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4241 - sex_out_loss: 0.4220 - age_out_loss: 9.0021 - val_loss: 12.3395 - val_sex_out_loss: 0.3767 - val_age_out_loss: 11.9628
Epoch 37/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.5010 - sex_out_loss: 0.4243 - age_out_loss: 9.0767 - val_loss: 15.2353 - val_sex_out_loss: 0.3842 - val_age_out_loss: 14.8511
Epoch 38/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4884 - sex_out_loss: 0.4186 - age_out_loss: 9.0697 - val_loss: 12.2915 - val_sex_out_loss: 0.3775 - val_age_out_loss: 11.9140
Epoch 39/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.2895 - sex_out_loss: 0.4112 - age_out_loss: 8.8783 - val_loss: 14.0048 - val_sex_out_loss: 0.3740 - val_age_out_loss: 13.6308
Epoch 40/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4554 - sex_out_loss: 0.4116 - age_out_loss: 9.0439 - val_loss: 15.8713 - val_sex_out_loss: 0.3755 - val_age_out_loss: 15.4958
Epoch 41/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.4379 - sex_out_loss: 0.4147 - age_out_loss: 9.0232 - val_loss: 11.9957 - val_sex_out_loss: 0.3685 - val_age_out_loss: 11.6272
Epoch 42/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 9.0477 - sex_out_loss: 0.4107 - age_out_loss: 8.6370 - val_loss: 11.8257 - val_sex_out_loss: 0.3793 - val_age_out_loss: 11.4465
Epoch 43/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.7804 - sex_out_loss: 0.4115 - age_out_loss: 8.3689 - val_loss: 10.9773 - val_sex_out_loss: 0.3712 - val_age_out_loss: 10.6061
Epoch 44/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.6453 - sex_out_loss: 0.4014 - age_out_loss: 8.2439 - val_loss: 13.4497 - val_sex_out_loss: 0.3646 - val_age_out_loss: 13.0851
Epoch 45/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.9189 - sex_out_loss: 0.4057 - age_out_loss: 8.5132 - val_loss: 12.6693 - val_sex_out_loss: 0.3619 - val_age_out_loss: 12.3074
Epoch 46/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.8543 - sex_out_loss: 0.4008 - age_out_loss: 8.4534 - val_loss: 12.0197 - val_sex_out_loss: 0.3617 - val_age_out_loss: 11.6580
Epoch 47/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.9497 - sex_out_loss: 0.4009 - age_out_loss: 8.5488 - val_loss: 11.3023 - val_sex_out_loss: 0.3554 - val_age_out_loss: 10.9469
Epoch 48/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.6011 - sex_out_loss: 0.3998 - age_out_loss: 8.2013 - val_loss: 11.8083 - val_sex_out_loss: 0.3651 - val_age_out_loss: 11.4431
Epoch 49/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.7430 - sex_out_loss: 0.4031 - age_out_loss: 8.3399 - val_loss: 10.4842 - val_sex_out_loss: 0.3566 - val_age_out_loss: 10.1276
Epoch 50/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.8291 - sex_out_loss: 0.3924 - age_out_loss: 8.4367 - val_loss: 12.2981 - val_sex_out_loss: 0.3593 - val_age_out_loss: 11.9387
Epoch 51/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.4162 - sex_out_loss: 0.3965 - age_out_loss: 8.0197 - val_loss: 11.4368 - val_sex_out_loss: 0.3497 - val_age_out_loss: 11.0871
Epoch 52/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.7157 - sex_out_loss: 0.3865 - age_out_loss: 8.3292 - val_loss: 10.6350 - val_sex_out_loss: 0.3594 - val_age_out_loss: 10.2756
Epoch 53/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.5031 - sex_out_loss: 0.3862 - age_out_loss: 8.1168 - val_loss: 11.2426 - val_sex_out_loss: 0.3511 - val_age_out_loss: 10.8915
Epoch 54/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.3442 - sex_out_loss: 0.3930 - age_out_loss: 7.9513 - val_loss: 12.4296 - val_sex_out_loss: 0.3558 - val_age_out_loss: 12.0738
Epoch 55/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.3584 - sex_out_loss: 0.3778 - age_out_loss: 7.9806 - val_loss: 12.9380 - val_sex_out_loss: 0.3565 - val_age_out_loss: 12.5815
Epoch 56/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.6795 - sex_out_loss: 0.3923 - age_out_loss: 8.2871 - val_loss: 12.6082 - val_sex_out_loss: 0.3538 - val_age_out_loss: 12.2544
Epoch 57/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.5503 - sex_out_loss: 0.3781 - age_out_loss: 8.1722 - val_loss: 13.1607 - val_sex_out_loss: 0.3503 - val_age_out_loss: 12.8104
Epoch 58/2000
```

```
6700/6700 [==============================] - 24s 4ms/step - loss: 8.2032 - sex_out_loss: 0.3849 - age_out_loss: 7.8183 - val_loss: 10.4161 - val_sex_out_loss: 0.3638 - val_age_out_loss: 10.0523
Epoch 59/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.1588 - sex_out_loss: 0.3808 - age_out_loss: 7.7779 - val_loss: 9.1041 - val_sex_out_loss: 0.3372 - val_age_out_loss: 8.7669
Epoch 60/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.3384 - sex_out_loss: 0.3710 - age_out_loss: 7.9674 - val_loss: 10.3142 - val_sex_out_loss: 0.3643 - val_age_out_loss: 9.9500
Epoch 61/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.2738 - sex_out_loss: 0.3920 - age_out_loss: 7.8818 - val_loss: 11.4123 - val_sex_out_loss: 0.3507 - val_age_out_loss: 11.0616
Epoch 62/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.3028 - sex_out_loss: 0.3859 - age_out_loss: 7.9169 - val_loss: 9.7152 - val_sex_out_loss: 0.3411 - val_age_out_loss: 9.3741
Epoch 63/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.9730 - sex_out_loss: 0.3782 - age_out_loss: 7.5948 - val_loss: 10.5901 - val_sex_out_loss: 0.3422 - val_age_out_loss: 10.2479
Epoch 64/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 8.0695 - sex_out_loss: 0.3774 - age_out_loss: 7.6921 - val_loss: 11.1465 - val_sex_out_loss: 0.3613 - val_age_out_loss: 10.7852
Epoch 65/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.0463 - sex_out_loss: 0.3891 - age_out_loss: 7.6572 - val_loss: 12.0078 - val_sex_out_loss: 0.3676 - val_age_out_loss: 11.6401
Epoch 66/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.0256 - sex_out_loss: 0.3711 - age_out_loss: 7.6546 - val_loss: 11.2449 - val_sex_out_loss: 0.3392 - val_age_out_loss: 10.9057
Epoch 67/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.7734 - sex_out_loss: 0.3748 - age_out_loss: 7.3985 - val_loss: 9.9378 - val_sex_out_loss: 0.3418 - val_age_out_loss: 9.5961
Epoch 68/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6671 - sex_out_loss: 0.3762 - age_out_loss: 7.2909 - val_loss: 10.9024 - val_sex_out_loss: 0.3406 - val_age_out_loss: 10.5618
Epoch 69/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.5585 - sex_out_loss: 0.3823 - age_out_loss: 8.1763 - val_loss: 13.2991 - val_sex_out_loss: 0.3607 - val_age_out_loss: 12.9384
Epoch 70/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.8815 - sex_out_loss: 0.3771 - age_out_loss: 7.5044 - val_loss: 10.0928 - val_sex_out_loss: 0.3356 - val_age_out_loss: 9.7572
Epoch 71/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6604 - sex_out_loss: 0.3652 - age_out_loss: 7.2952 - val_loss: 9.9195 - val_sex_out_loss: 0.3371 - val_age_out_loss: 9.5825
Epoch 72/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6632 - sex_out_loss: 0.3650 - age_out_loss: 7.2982 - val_loss: 10.7414 - val_sex_out_loss: 0.3335 - val_age_out_loss: 10.4078
Epoch 73/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.7693 - sex_out_loss: 0.3635 - age_out_loss: 7.4058 - val_loss: 11.4377 - val_sex_out_loss: 0.3452 - val_age_out_loss: 11.0925
Epoch 74/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.0074 - sex_out_loss: 0.3742 - age_out_loss: 7.6332 - val_loss: 8.7869 - val_sex_out_loss: 0.3449 - val_age_out_loss: 8.4420
Epoch 75/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.7302 - sex_out_loss: 0.3659 - age_out_loss: 7.3644 - val_loss: 10.5687 - val_sex_out_loss: 0.3630 - val_age_out_loss: 10.2057
Epoch 76/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6391 - sex_out_loss: 0.3638 - age_out_loss: 7.2753 - val_loss: 9.4072 - val_sex_out_loss: 0.3374 - val_age_out_loss: 9.0698
Epoch 77/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6725 - sex_out_loss: 0.3717 - age_out_loss: 7.3008 - val_loss: 12.8748 - val_sex_out_loss: 0.3395 - val_age_out_loss: 12.5353
Epoch 78/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.8447 - sex_out_loss: 0.3719 - age_out_loss: 7.4729 - val_loss: 10.8017 - val_sex_out_loss: 0.3403 - val_age_out_loss: 10.4614
Epoch 79/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.5698 - sex_out_loss: 0.3654 - age_out_loss: 7.2044 - val_loss: 13.1882 - val_sex_out_loss: 0.3389 - val_age_out_loss: 12.8493
Epoch 80/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 8.0342 - sex_out_loss: 0.3628 - age_out_loss: 7.6714 - val_loss: 12.6290 - val_sex_out_loss: 0.3561 - val_age_out_loss: 12.2729
Epoch 81/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6178 - sex_out_loss: 0.3564 - age_out_loss: 7.2613 - val_loss: 9.1046 - val_sex_out_loss: 0.3379 - val_age_out_loss: 8.7668
Epoch 82/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.4460 - sex_out_loss: 0.3534 - age_out_loss: 7.0926 - val_loss: 9.9027 - val_sex_out_loss: 0.3363 - val_age_out_loss: 9.5664
Epoch 83/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.6954 - sex_out_loss: 0.3615 - age_out_loss: 7.3339 - val_loss: 10.6613 - val_sex_out_loss: 0.3389 - val_age_out_loss: 10.3224
Epoch 84/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.4494 - sex_out_loss: 0.3602 - age_out_loss: 7.0892 - val_loss: 10.6077 - val_sex_out_loss: 0.3484 - val_age_out_loss: 10.2593
Epoch 85/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.4321 - sex_out_loss: 0.3526 - age_out_loss: 7.0795 - val_loss: 10.9336 - val_sex_out_loss: 0.3387 - val_age_out_loss: 10.5949
Epoch 86/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2130 - sex_out_loss: 0.3517 - age_out_loss: 6.8613 - val_loss: 9.5036 - val_sex_out_loss: 0.3239 - val_age_out_loss: 9.1797
Epoch 87/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.8723 - sex_out_loss: 0.3590 - age_out_loss: 7.5133 - val_loss: 11.5854 - val_sex_out_loss: 0.3311 - val_age_out_loss: 11.2542
Epoch 88/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1119 - sex_out_loss: 0.3471 - age_out_loss: 6.7648 - val_loss: 9.7643 - val_sex_out_loss: 0.3233 - val_age_out_loss: 9.4410
Epoch 89/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.5391 - sex_out_loss: 0.3491 - age_out_loss: 7.1900 - val_loss: 11.8398 - val_sex_out_loss: 0.3264 - val_age_out_loss: 11.5134
Epoch 90/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2079 - sex_out_loss: 0.3451 - age_out_loss: 6.8628 - val_loss: 11.2559 - val_sex_out_loss: 0.3337 - val_age_out_loss: 10.9222
Epoch 91/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2443 - sex_out_loss: 0.3491 - age_out_loss: 6.8952 - val_loss: 10.6180 - val_sex_out_loss: 0.3340 - val_age_out_loss: 10.2840
Epoch 92/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1968 - sex_out_loss: 0.3523 - age_out_loss: 6.8445 - val_loss: 8.5158 - val_sex_out_loss: 0.3205 - val_age_out_loss: 8.1953
Epoch 93/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1464 - sex_out_loss: 0.3436 - age_out_loss: 6.8028 - val_loss: 10.5327 - val_sex_out_loss: 0.3194 - val_age_out_loss: 10.2133
Epoch 94/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1297 - sex_out_loss: 0.3371 - age_out_loss: 6.7926 - val_loss: 10.4581 - val_sex_out_loss: 0.3256 - val_age_out_loss: 10.1325
Epoch 95/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2035 - sex_out_loss: 0.3417 - age_out_loss: 6.8618 - val_loss: 11.6116 - val_sex_out_loss: 0.3256 - val_age_out_loss: 11.2860
Epoch 96/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2762 - sex_out_loss: 0.3443 - age_out_loss: 6.9318 - val_loss: 10.1277 - val_sex_out_loss: 0.3183 - val_age_out_loss: 9.8094
Epoch 97/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.0188 - sex_out_loss: 0.3351 - age_out_loss: 6.6836 - val_loss: 13.0182 - val_sex_out_loss: 0.3352 - val_age_out_loss: 12.6830
Epoch 98/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.2908 - sex_out_loss: 0.3416 - age_out_loss: 6.9492 - val_loss: 9.9394 - val_sex_out_loss: 0.3169 - val_age_out_loss: 9.6225
Epoch 99/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.4279 - sex_out_loss: 0.3482 - age_out_loss: 7.0796 - val_loss: 9.9889 - val_sex_out_loss: 0.3365 - val_age_out_loss: 9.6524
Epoch 100/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1155 - sex_out_loss: 0.3493 - age_out_loss: 6.7662 - val_loss: 9.3770 - val_sex_out_loss: 0.3256 - val_age_out_loss: 9.0514
Epoch 101/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1202 - sex_out_loss: 0.3447 - age_out_loss: 6.7755 - val_loss: 10.2106 - val_sex_out_loss: 0.3339 - val_age_out_loss: 9.8768
Epoch 102/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 7.1853 - sex_out_loss: 0.3378 - age_out_loss: 6.8475 - val_loss: 9.1386 - val_sex_out_loss: 0.3179 - val_age_out_loss: 8.8207
Epoch 103/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.9143 - sex_out_loss: 0.3455 - age_out_loss: 6.5688 - val_loss: 9.7937 - val_sex_out_loss: 0.3232 - val_age_out_loss: 9.4705
Epoch 104/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.9190 - sex_out_loss: 0.3379 - age_out_loss: 6.5812 - val_loss: 11.2118 - val_sex_out_loss: 0.3231 - val_age_out_loss: 10.8887
Epoch 105/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.0477 - sex_out_loss: 0.3385 - age_out_loss: 6.7093 - val_loss: 8.7853 - val_sex_out_loss: 0.3278 - val_age_out_loss: 8.4576
Epoch 106/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8773 - sex_out_loss: 0.3390 - age_out_loss: 6.5382 - val_loss: 9.3295 - val_sex_out_loss: 0.3213 - val_age_out_loss: 9.0082
Epoch 107/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.1609 - sex_out_loss: 0.3418 - age_out_loss: 6.8191 - val_loss: 10.7503 - val_sex_out_loss: 0.3232 - val_age_out_loss: 10.4271
Epoch 108/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 7.2777 - sex_out_loss: 0.3348 - age_out_loss: 6.9429 - val_loss: 9.6964 - val_sex_out_loss: 0.3194 - val_age_out_loss: 9.3770
Epoch 109/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8927 - sex_out_loss: 0.3368 - age_out_loss: 6.5559 - val_loss: 9.2511 - val_sex_out_loss: 0.3188 - val_age_out_loss: 8.9323
Epoch 110/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.9711 - sex_out_loss: 0.3308 - age_out_loss: 6.6403 - val_loss: 9.4564 - val_sex_out_loss: 0.3235 - val_age_out_loss: 9.1329
Epoch 111/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8539 - sex_out_loss: 0.3280 - age_out_loss: 6.5259 - val_loss: 9.3505 - val_sex_out_loss: 0.3202 - val_age_out_loss: 9.0303
Epoch 112/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.7543 - sex_out_loss: 0.3336 - age_out_loss: 6.4207 - val_loss: 9.6400 - val_sex_out_loss: 0.3260 - val_age_out_loss: 9.3140
Epoch 113/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6906 - sex_out_loss: 0.3297 - age_out_loss: 6.3608 - val_loss: 9.4699 - val_sex_out_loss: 0.3143 - val_age_out_loss: 9.1556
Epoch 114/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8903 - sex_out_loss: 0.3398 - age_out_loss: 6.5504 - val_loss: 9.4991 - val_sex_out_loss: 0.3174 - val_age_out_loss: 9.1817
Epoch 115/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6574 - sex_out_loss: 0.3315 - age_out_loss: 6.3259 - val_loss: 8.9596 - val_sex_out_loss: 0.3168 - val_age_out_loss: 8.6429
Epoch 116/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6515 - sex_out_loss: 0.3354 - age_out_loss: 6.3160 - val_loss: 9.7221 - val_sex_out_loss: 0.3214 - val_age_out_loss: 9.4007
Epoch 117/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.9097 - sex_out_loss: 0.3376 - age_out_loss: 6.5721 - val_loss: 11.3239 - val_sex_out_loss: 0.3250 - val_age_out_loss: 10.9989
Epoch 118/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8698 - sex_out_loss: 0.3299 - age_out_loss: 6.5400 - val_loss: 8.9914 - val_sex_out_loss: 0.3214 - val_age_out_loss: 8.6700
Epoch 119/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4710 - sex_out_loss: 0.3225 - age_out_loss: 6.1485 - val_loss: 9.1091 - val_sex_out_loss: 0.3171 - val_age_out_loss: 8.7919
Epoch 120/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.7556 - sex_out_loss: 0.3358 - age_out_loss: 6.4197 - val_loss: 9.2330 - val_sex_out_loss: 0.3217 - val_age_out_loss: 8.9114
Epoch 121/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6628 - sex_out_loss: 0.3244 - age_out_loss: 6.3384 - val_loss: 9.9459 - val_sex_out_loss: 0.3178 - val_age_out_loss: 9.6281
Epoch 122/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4575 - sex_out_loss: 0.3238 - age_out_loss: 6.1337 - val_loss: 8.8944 - val_sex_out_loss: 0.3193 - val_age_out_loss: 8.5750
Epoch 123/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5943 - sex_out_loss: 0.3132 - age_out_loss: 6.2811 - val_loss: 8.9810 - val_sex_out_loss: 0.3184 - val_age_out_loss: 8.6626
Epoch 124/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8669 - sex_out_loss: 0.3335 - age_out_loss: 6.5335 - val_loss: 10.7887 - val_sex_out_loss: 0.3249 - val_age_out_loss: 10.4638
Epoch 125/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.9425 - sex_out_loss: 0.3197 - age_out_loss: 6.6228 - val_loss: 9.7569 - val_sex_out_loss: 0.3346 - val_age_out_loss: 9.4222
Epoch 126/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.8821 - sex_out_loss: 0.3411 - age_out_loss: 6.5410 - val_loss: 10.4065 - val_sex_out_loss: 0.3263 - val_age_out_loss: 10.0802
Epoch 127/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5618 - sex_out_loss: 0.3423 - age_out_loss: 6.2196 - val_loss: 8.9021 - val_sex_out_loss: 0.3187 - val_age_out_loss: 8.5835
Epoch 128/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6089 - sex_out_loss: 0.3253 - age_out_loss: 6.2835 - val_loss: 9.3954 - val_sex_out_loss: 0.3218 - val_age_out_loss: 9.0736
Epoch 129/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6379 - sex_out_loss: 0.3338 - age_out_loss: 6.3041 - val_loss: 8.6590 - val_sex_out_loss: 0.3128 - val_age_out_loss: 8.3461
```

```
Epoch 130/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5489 - sex_out_loss: 0.3295 - age_out_loss: 6.2194 - val_loss: 9.5612 - val_sex_out_loss: 0.3152 - val_age_out_loss: 9.2460
Epoch 131/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3057 - sex_out_loss: 0.3187 - age_out_loss: 5.9870 - val_loss: 10.1530 - val_sex_out_loss: 0.3213 - val_age_out_loss: 9.8318
Epoch 132/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5096 - sex_out_loss: 0.3264 - age_out_loss: 6.1832 - val_loss: 9.0971 - val_sex_out_loss: 0.3225 - val_age_out_loss: 8.7745
Epoch 133/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4632 - sex_out_loss: 0.3253 - age_out_loss: 6.1379 - val_loss: 9.1982 - val_sex_out_loss: 0.3247 - val_age_out_loss: 8.8735
Epoch 134/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.7202 - sex_out_loss: 0.3326 - age_out_loss: 6.3876 - val_loss: 12.2550 - val_sex_out_loss: 0.3344 - val_age_out_loss: 11.9206
Epoch 135/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 7.0224 - sex_out_loss: 0.3300 - age_out_loss: 6.6924 - val_loss: 8.4047 - val_sex_out_loss: 0.3282 - val_age_out_loss: 8.0765
Epoch 136/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6776 - sex_out_loss: 0.3278 - age_out_loss: 6.3498 - val_loss: 8.3702 - val_sex_out_loss: 0.3135 - val_age_out_loss: 8.0567
Epoch 137/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5546 - sex_out_loss: 0.3193 - age_out_loss: 6.2352 - val_loss: 8.4641 - val_sex_out_loss: 0.3179 - val_age_out_loss: 8.1462
Epoch 138/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3581 - sex_out_loss: 0.3223 - age_out_loss: 6.0358 - val_loss: 9.6722 - val_sex_out_loss: 0.3243 - val_age_out_loss: 9.3479
Epoch 139/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3651 - sex_out_loss: 0.3152 - age_out_loss: 6.0499 - val_loss: 9.1594 - val_sex_out_loss: 0.3217 - val_age_out_loss: 8.8376
Epoch 140/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2404 - sex_out_loss: 0.3205 - age_out_loss: 5.9199 - val_loss: 10.2623 - val_sex_out_loss: 0.3173 - val_age_out_loss: 9.9450
Epoch 141/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6022 - sex_out_loss: 0.3233 - age_out_loss: 6.2788 - val_loss: 9.4858 - val_sex_out_loss: 0.3169 - val_age_out_loss: 9.1688
Epoch 142/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5183 - sex_out_loss: 0.3147 - age_out_loss: 6.2036 - val_loss: 9.1453 - val_sex_out_loss: 0.3169 - val_age_out_loss: 8.8284
Epoch 143/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.6283 - sex_out_loss: 0.3227 - age_out_loss: 6.3056 - val_loss: 8.4189 - val_sex_out_loss: 0.3200 - val_age_out_loss: 8.0989
Epoch 144/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2321 - sex_out_loss: 0.3245 - age_out_loss: 5.9075 - val_loss: 9.8073 - val_sex_out_loss: 0.3185 - val_age_out_loss: 9.4888
Epoch 145/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.5972 - sex_out_loss: 0.3224 - age_out_loss: 6.2749 - val_loss: 10.3252 - val_sex_out_loss: 0.3084 - val_age_out_loss: 10.0168
Epoch 146/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3054 - sex_out_loss: 0.3116 - age_out_loss: 5.9937 - val_loss: 9.5511 - val_sex_out_loss: 0.3246 - val_age_out_loss: 9.2265
Epoch 147/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3784 - sex_out_loss: 0.3200 - age_out_loss: 6.0583 - val_loss: 9.4944 - val_sex_out_loss: 0.3172 - val_age_out_loss: 9.1772
Epoch 148/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2309 - sex_out_loss: 0.3172 - age_out_loss: 5.9137 - val_loss: 9.2794 - val_sex_out_loss: 0.3087 - val_age_out_loss: 8.9708
Epoch 149/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4780 - sex_out_loss: 0.3118 - age_out_loss: 6.1662 - val_loss: 8.5069 - val_sex_out_loss: 0.3116 - val_age_out_loss: 8.1952
Epoch 150/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3281 - sex_out_loss: 0.3161 - age_out_loss: 6.0120 - val_loss: 8.3953 - val_sex_out_loss: 0.3189 - val_age_out_loss: 8.0764
Epoch 151/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1356 - sex_out_loss: 0.3096 - age_out_loss: 5.8261 - val_loss: 9.1160 - val_sex_out_loss: 0.3324 - val_age_out_loss: 8.7836
Epoch 152/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2463 - sex_out_loss: 0.3208 - age_out_loss: 5.9255 - val_loss: 8.5615 - val_sex_out_loss: 0.3113 - val_age_out_loss: 8.2502
Epoch 153/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2199 - sex_out_loss: 0.3148 - age_out_loss: 5.9051 - val_loss: 9.1146 - val_sex_out_loss: 0.3079 - val_age_out_loss: 8.8068
Epoch 154/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0909 - sex_out_loss: 0.3162 - age_out_loss: 5.7747 - val_loss: 8.6110 - val_sex_out_loss: 0.3187 - val_age_out_loss: 8.2923
Epoch 155/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1874 - sex_out_loss: 0.3052 - age_out_loss: 5.8822 - val_loss: 8.6036 - val_sex_out_loss: 0.3209 - val_age_out_loss: 8.2827
Epoch 156/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3376 - sex_out_loss: 0.3175 - age_out_loss: 6.0201 - val_loss: 8.6741 - val_sex_out_loss: 0.3140 - val_age_out_loss: 8.3601
Epoch 157/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1695 - sex_out_loss: 0.3018 - age_out_loss: 5.8676 - val_loss: 9.5277 - val_sex_out_loss: 0.3118 - val_age_out_loss: 9.2159
Epoch 158/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3385 - sex_out_loss: 0.3106 - age_out_loss: 6.0280 - val_loss: 8.9666 - val_sex_out_loss: 0.3201 - val_age_out_loss: 8.6465
Epoch 159/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.3345 - sex_out_loss: 0.3072 - age_out_loss: 6.0274 - val_loss: 9.4705 - val_sex_out_loss: 0.3131 - val_age_out_loss: 9.1573
Epoch 160/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0980 - sex_out_loss: 0.3108 - age_out_loss: 5.7873 - val_loss: 9.6679 - val_sex_out_loss: 0.3108 - val_age_out_loss: 9.3571
Epoch 161/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1928 - sex_out_loss: 0.3063 - age_out_loss: 5.8865 - val_loss: 9.1921 - val_sex_out_loss: 0.3118 - val_age_out_loss: 8.8803
Epoch 162/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2469 - sex_out_loss: 0.3103 - age_out_loss: 5.9367 - val_loss: 8.1444 - val_sex_out_loss: 0.3113 - val_age_out_loss: 7.8332
Epoch 163/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1720 - sex_out_loss: 0.3095 - age_out_loss: 5.8625 - val_loss: 8.3270 - val_sex_out_loss: 0.3109 - val_age_out_loss: 8.0161
Epoch 164/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1281 - sex_out_loss: 0.3093 - age_out_loss: 5.8188 - val_loss: 8.6298 - val_sex_out_loss: 0.3115 - val_age_out_loss: 8.3183
Epoch 165/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0556 - sex_out_loss: 0.2976 - age_out_loss: 5.7580 - val_loss: 8.3925 - val_sex_out_loss: 0.3111 - val_age_out_loss: 8.0814
Epoch 166/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0182 - sex_out_loss: 0.3044 - age_out_loss: 5.7138 - val_loss: 9.2152 - val_sex_out_loss: 0.3070 - val_age_out_loss: 8.9082
Epoch 167/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0681 - sex_out_loss: 0.3074 - age_out_loss: 5.7607 - val_loss: 8.5895 - val_sex_out_loss: 0.3080 - val_age_out_loss: 8.2815
Epoch 168/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4695 - sex_out_loss: 0.3114 - age_out_loss: 6.1581 - val_loss: 9.5967 - val_sex_out_loss: 0.3105 - val_age_out_loss: 9.2862
Epoch 169/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2167 - sex_out_loss: 0.3112 - age_out_loss: 5.9055 - val_loss: 8.9847 - val_sex_out_loss: 0.3137 - val_age_out_loss: 8.6710
Epoch 170/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9801 - sex_out_loss: 0.3131 - age_out_loss: 5.6670 - val_loss: 9.3076 - val_sex_out_loss: 0.3118 - val_age_out_loss: 8.9957
Epoch 171/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0434 - sex_out_loss: 0.3062 - age_out_loss: 5.7372 - val_loss: 8.9323 - val_sex_out_loss: 0.3467 - val_age_out_loss: 8.5856
Epoch 172/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0482 - sex_out_loss: 0.3116 - age_out_loss: 5.7365 - val_loss: 8.9554 - val_sex_out_loss: 0.3098 - val_age_out_loss: 8.6455
Epoch 173/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0659 - sex_out_loss: 0.3091 - age_out_loss: 5.7568 - val_loss: 8.6235 - val_sex_out_loss: 0.3063 - val_age_out_loss: 8.3172
Epoch 174/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0009 - sex_out_loss: 0.3031 - age_out_loss: 5.6978 - val_loss: 8.4886 - val_sex_out_loss: 0.3082 - val_age_out_loss: 8.1805
Epoch 175/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0917 - sex_out_loss: 0.3000 - age_out_loss: 5.7917 - val_loss: 8.3392 - val_sex_out_loss: 0.3060 - val_age_out_loss: 8.0331
Epoch 176/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.4278 - sex_out_loss: 0.3156 - age_out_loss: 6.1122 - val_loss: 10.7049 - val_sex_out_loss: 0.3151 - val_age_out_loss: 10.3897
Epoch 177/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 6.1106 - sex_out_loss: 0.3028 - age_out_loss: 5.8078 - val_loss: 8.9057 - val_sex_out_loss: 0.3074 - val_age_out_loss: 8.5983
Epoch 178/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8236 - sex_out_loss: 0.2926 - age_out_loss: 5.5310 - val_loss: 8.6758 - val_sex_out_loss: 0.3092 - val_age_out_loss: 8.3666
Epoch 179/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0934 - sex_out_loss: 0.3057 - age_out_loss: 5.7877 - val_loss: 9.4421 - val_sex_out_loss: 0.3070 - val_age_out_loss: 9.1350
Epoch 180/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9894 - sex_out_loss: 0.3066 - age_out_loss: 5.6828 - val_loss: 9.8110 - val_sex_out_loss: 0.3111 - val_age_out_loss: 9.5000
Epoch 181/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0283 - sex_out_loss: 0.3025 - age_out_loss: 5.7258 - val_loss: 8.1747 - val_sex_out_loss: 0.3004 - val_age_out_loss: 7.8744
Epoch 182/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1151 - sex_out_loss: 0.3066 - age_out_loss: 5.8084 - val_loss: 8.5955 - val_sex_out_loss: 0.3055 - val_age_out_loss: 8.2899
Epoch 183/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0076 - sex_out_loss: 0.3016 - age_out_loss: 5.7060 - val_loss: 8.5091 - val_sex_out_loss: 0.3199 - val_age_out_loss: 8.1892
Epoch 184/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1348 - sex_out_loss: 0.2946 - age_out_loss: 5.8402 - val_loss: 10.5669 - val_sex_out_loss: 0.3038 - val_age_out_loss: 10.2630
Epoch 185/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1220 - sex_out_loss: 0.3079 - age_out_loss: 5.8140 - val_loss: 8.2390 - val_sex_out_loss: 0.3114 - val_age_out_loss: 7.9275
Epoch 186/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0454 - sex_out_loss: 0.3077 - age_out_loss: 5.7376 - val_loss: 9.0537 - val_sex_out_loss: 0.3124 - val_age_out_loss: 8.7413
Epoch 187/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1442 - sex_out_loss: 0.3031 - age_out_loss: 5.8411 - val_loss: 8.4620 - val_sex_out_loss: 0.3221 - val_age_out_loss: 8.1399
Epoch 188/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2038 - sex_out_loss: 0.3048 - age_out_loss: 5.8990 - val_loss: 9.3889 - val_sex_out_loss: 0.3112 - val_age_out_loss: 9.0777
Epoch 189/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8700 - sex_out_loss: 0.3006 - age_out_loss: 5.5693 - val_loss: 10.1410 - val_sex_out_loss: 0.3079 - val_age_out_loss: 9.8331
Epoch 190/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8559 - sex_out_loss: 0.3060 - age_out_loss: 5.5499 - val_loss: 8.7629 - val_sex_out_loss: 0.3122 - val_age_out_loss: 8.4507
Epoch 191/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8253 - sex_out_loss: 0.2917 - age_out_loss: 5.5336 - val_loss: 9.0901 - val_sex_out_loss: 0.3091 - val_age_out_loss: 8.7810
Epoch 192/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1054 - sex_out_loss: 0.2942 - age_out_loss: 5.8113 - val_loss: 8.4340 - val_sex_out_loss: 0.3112 - val_age_out_loss: 8.1228
Epoch 193/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9840 - sex_out_loss: 0.2919 - age_out_loss: 5.6921 - val_loss: 8.4172 - val_sex_out_loss: 0.3027 - val_age_out_loss: 8.1145
Epoch 194/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9032 - sex_out_loss: 0.3033 - age_out_loss: 5.5999 - val_loss: 10.3781 - val_sex_out_loss: 0.3351 - val_age_out_loss: 10.0431
Epoch 195/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.2441 - sex_out_loss: 0.2954 - age_out_loss: 5.9488 - val_loss: 9.3498 - val_sex_out_loss: 0.3105 - val_age_out_loss: 9.0394
Epoch 196/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1991 - sex_out_loss: 0.3057 - age_out_loss: 5.8935 - val_loss: 8.4474 - val_sex_out_loss: 0.3146 - val_age_out_loss: 8.1328
Epoch 197/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.0763 - sex_out_loss: 0.3098 - age_out_loss: 5.7665 - val_loss: 9.2721 - val_sex_out_loss: 0.3088 - val_age_out_loss: 8.9633
Epoch 198/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8810 - sex_out_loss: 0.2953 - age_out_loss: 5.5857 - val_loss: 8.5649 - val_sex_out_loss: 0.3149 - val_age_out_loss: 8.2500
Epoch 199/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9225 - sex_out_loss: 0.2887 - age_out_loss: 5.6339 - val_loss: 8.4396 - val_sex_out_loss: 0.3047 - val_age_out_loss: 8.1349
Epoch 200/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9025 - sex_out_loss: 0.2907 - age_out_loss: 5.6118 - val_loss: 8.6911 - val_sex_out_loss: 0.3010 - val_age_out_loss: 8.3901
Epoch 201/2000
```

```
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9062 - sex_out_loss: 0.2983 - age_out_loss: 5.6079 - val_loss: 8.7387 - val_sex_out_loss: 0.3062 - val_age_out_loss: 8.4324
Epoch 202/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7764 - sex_out_loss: 0.2861 - age_out_loss: 5.4903 - val_loss: 8.6256 - val_sex_out_loss: 0.3047 - val_age_out_loss: 8.3208
Epoch 203/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8583 - sex_out_loss: 0.2975 - age_out_loss: 5.5607 - val_loss: 8.5594 - val_sex_out_loss: 0.3093 - val_age_out_loss: 8.2501
Epoch 204/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7230 - sex_out_loss: 0.2892 - age_out_loss: 5.4338 - val_loss: 8.7623 - val_sex_out_loss: 0.3031 - val_age_out_loss: 8.4591
Epoch 205/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8094 - sex_out_loss: 0.2904 - age_out_loss: 5.5190 - val_loss: 8.5753 - val_sex_out_loss: 0.3079 - val_age_out_loss: 8.2674
Epoch 206/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8841 - sex_out_loss: 0.2938 - age_out_loss: 5.5903 - val_loss: 9.1823 - val_sex_out_loss: 0.3047 - val_age_out_loss: 8.8776
Epoch 207/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9405 - sex_out_loss: 0.2988 - age_out_loss: 5.6418 - val_loss: 9.6637 - val_sex_out_loss: 0.3171 - val_age_out_loss: 9.3466
Epoch 208/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9155 - sex_out_loss: 0.3030 - age_out_loss: 5.6124 - val_loss: 9.3329 - val_sex_out_loss: 0.3089 - val_age_out_loss: 9.0240
Epoch 209/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7637 - sex_out_loss: 0.2918 - age_out_loss: 5.4719 - val_loss: 8.5213 - val_sex_out_loss: 0.3104 - val_age_out_loss: 8.2109
Epoch 210/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7730 - sex_out_loss: 0.2923 - age_out_loss: 5.4807 - val_loss: 9.3010 - val_sex_out_loss: 0.2979 - val_age_out_loss: 9.0031
Epoch 211/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 6.1829 - sex_out_loss: 0.2946 - age_out_loss: 5.8882 - val_loss: 8.3690 - val_sex_out_loss: 0.3028 - val_age_out_loss: 8.0662
Epoch 212/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 5.8453 - sex_out_loss: 0.2899 - age_out_loss: 5.5554 - val_loss: 8.9949 - val_sex_out_loss: 0.3052 - val_age_out_loss: 8.6897
Epoch 213/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9105 - sex_out_loss: 0.2896 - age_out_loss: 5.6209 - val_loss: 8.6946 - val_sex_out_loss: 0.3038 - val_age_out_loss: 8.3908
Epoch 214/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8250 - sex_out_loss: 0.2939 - age_out_loss: 5.5312 - val_loss: 9.9556 - val_sex_out_loss: 0.3070 - val_age_out_loss: 9.6485
Epoch 215/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 5.9298 - sex_out_loss: 0.2874 - age_out_loss: 5.6424 - val_loss: 8.6697 - val_sex_out_loss: 0.2992 - val_age_out_loss: 8.3705
Epoch 216/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7362 - sex_out_loss: 0.2868 - age_out_loss: 5.4494 - val_loss: 8.7560 - val_sex_out_loss: 0.3083 - val_age_out_loss: 8.4477
Epoch 217/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8300 - sex_out_loss: 0.2904 - age_out_loss: 5.5396 - val_loss: 8.7738 - val_sex_out_loss: 0.3011 - val_age_out_loss: 8.4726
Epoch 218/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8391 - sex_out_loss: 0.2835 - age_out_loss: 5.5556 - val_loss: 8.6144 - val_sex_out_loss: 0.3119 - val_age_out_loss: 8.3025
Epoch 219/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7101 - sex_out_loss: 0.2907 - age_out_loss: 5.4194 - val_loss: 8.7287 - val_sex_out_loss: 0.3025 - val_age_out_loss: 8.4262
Epoch 220/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6020 - sex_out_loss: 0.2868 - age_out_loss: 5.3152 - val_loss: 9.3843 - val_sex_out_loss: 0.3121 - val_age_out_loss: 9.0722
Epoch 221/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.8314 - sex_out_loss: 0.2831 - age_out_loss: 5.5483 - val_loss: 8.4830 - val_sex_out_loss: 0.3134 - val_age_out_loss: 8.1696
Epoch 222/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.5988 - sex_out_loss: 0.2787 - age_out_loss: 5.3200 - val_loss: 8.7964 - val_sex_out_loss: 0.3002 - val_age_out_loss: 8.4962
Epoch 223/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.5596 - sex_out_loss: 0.2804 - age_out_loss: 5.2792 - val_loss: 9.0093 - val_sex_out_loss: 0.3116 - val_age_out_loss: 8.6977
Epoch 224/2000
6700/6700 [==============================] - 25s 4ms/step - loss: 5.8659 - sex_out_loss: 0.2825 - age_out_loss: 5.5834 - val_loss: 8.1877 - val_sex_out_loss: 0.2957 - val_age_out_loss: 7.8920
Epoch 225/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6578 - sex_out_loss: 0.2787 - age_out_loss: 5.3790 - val_loss: 9.1938 - val_sex_out_loss: 0.3051 - val_age_out_loss: 8.8887
Epoch 226/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6644 - sex_out_loss: 0.2840 - age_out_loss: 5.3804 - val_loss: 8.4783 - val_sex_out_loss: 0.3000 - val_age_out_loss: 8.1783
Epoch 227/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6403 - sex_out_loss: 0.2846 - age_out_loss: 5.3556 - val_loss: 8.9956 - val_sex_out_loss: 0.3020 - val_age_out_loss: 8.6936
Epoch 228/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7196 - sex_out_loss: 0.2974 - age_out_loss: 5.4222 - val_loss: 8.3669 - val_sex_out_loss: 0.2998 - val_age_out_loss: 8.0671
Epoch 229/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7357 - sex_out_loss: 0.2861 - age_out_loss: 5.4496 - val_loss: 8.7744 - val_sex_out_loss: 0.3057 - val_age_out_loss: 8.4687
Epoch 230/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6449 - sex_out_loss: 0.2897 - age_out_loss: 5.3552 - val_loss: 8.9416 - val_sex_out_loss: 0.2938 - val_age_out_loss: 8.6479
Epoch 231/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6904 - sex_out_loss: 0.2797 - age_out_loss: 5.4107 - val_loss: 9.3288 - val_sex_out_loss: 0.2973 - val_age_out_loss: 9.0315
Epoch 232/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.4429 - sex_out_loss: 0.2883 - age_out_loss: 5.1546 - val_loss: 8.4817 - val_sex_out_loss: 0.3065 - val_age_out_loss: 8.1752
Epoch 233/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6943 - sex_out_loss: 0.2857 - age_out_loss: 5.4087 - val_loss: 8.9663 - val_sex_out_loss: 0.2993 - val_age_out_loss: 8.6670
Epoch 234/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.7333 - sex_out_loss: 0.2871 - age_out_loss: 5.4463 - val_loss: 8.3461 - val_sex_out_loss: 0.2973 - val_age_out_loss: 8.0488
Epoch 235/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.6417 - sex_out_loss: 0.2774 - age_out_loss: 5.3643 - val_loss: 9.2349 - val_sex_out_loss: 0.2987 - val_age_out_loss: 8.9362
Epoch 236/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9022 - sex_out_loss: 0.2911 - age_out_loss: 5.6111 - val_loss: 9.2097 - val_sex_out_loss: 0.3005 - val_age_out_loss: 8.9092
Epoch 237/2000
6700/6700 [==============================] - 24s 4ms/step - loss: 5.9485 - sex_out_loss: 0.2914 - age_out_loss: 5.6570 - val_loss: 8.5123 - val_sex_out_loss: 0.2949 - val_age_out_loss: 8.2174
<keras.callbacks.History at 0x7f1a2ad42860
```

4.2.4 Saving the model

```
model.save('drive/My Drive/23Age_Gender_model.h5')
```

4.3 Web View

4.3.1 Front Page

```
<!DOCTYPE html>
{% load static %}
<html lang="en">

<head>
<meta charset="utf-8">
<title>Vision</title>
<meta content="width=device-width, initial-scale=1.0"
name="viewport">
<meta content="" name="keywords">
<meta content="" name="description">

<!-- Favicons -->
<link href="{% static 'ing/favicon.jpg' %}" rel="icon">


<!-- Google Fonts -->
<link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i
,400,400i,700,700i|Raleway:300,400,500,700,800"
rel="stylesheet">

<!-- Bootstrap CSS File -->
<link href="{% static 'lib/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">

<!-- Libraries CSS Files -->
<link href="{% static 'lib/font-awesome/css/font-
awesome.min.css' %}" rel="stylesheet">
<link rel="stylesheet" href="{% static
'lib/animate/animate.min.css' %}" >
<link href="{% static 'lib/venobox/venobox.css' %}"
rel="stylesheet">
<link href="{% static
'lib/owlcarousel/assets/owl.carousel.min.css' %}"
rel="stylesheet">

<!-- Main Stylesheet File -->
<link href="{% static 'css/style.css' %}" rel="stylesheet">
<!-- =======================================================
Theme Name: TheEvent
Theme URL: https://bootstrapmade.com/theevent-conference-event-
bootstrap-template/
```

```
Author: BootstrapMade.com
License: https://bootstrapmade.com/license/
</head>

<body>

<!--===================Header ============================ >
<header id="header">
<div class="container">

<div id="logo" class="pull-left text-light">
<!-- Uncomment below if you prefer to use a text logo -->
<!-- <h1><a href="#main">C<span>o</span>nf</a></h1>-->
<img src="static/img/logo (2).png" alt="" title="">Vision
</div>

<nav id="nav-menu-container">
<ul class="nav-menu">
<li class="menu-active"><a href="">Home</a></li>
</ul>
</nav><!-- #nav-menu-container -->
</div>
</header><!-- #header -->

<!--===================== Intro Section =====================-->
<section id="intro">
<div class="intro-container wow fadeIn">
<h1 class="mb-4 pb-0">Know Your <span>Age</span> Here</h1>

<! ===================  Camera opener========================>

<button class=" play-btn mb-4" data-toggle="modal" data-
target="#modelId" onclick="cam()"></button>
<!-- Modal -->
<div class="modal fade" id="modelId" tabindex="-1" role="dialog"
aria-labelledby="modelTitleId"
aria-hidden="true">
<div class="modal-dialog modal-lg" role="document">
<div class="modal-content">
<div class="modal-body">
<div class="card border-0 ">
<div class="card-body">
<video id="videoElement"  autoplay="true"  width="680px"
height="auto">
no video support......
</video>
<button class="close" data-dismiss="modal" aria-label="Close">
```

```
<i class="fa fa-times" aria-hidden="true"></i>
</button>
</div>
</div>
</div>
</div>
</div>
</div>

<! ================== Subscribe Button =====================>

<a href="suscribe/" class="about-btn scrollto">Subscribe</a>
{% csrf_token %}
<form action="pic_upload/" method="POST" enctype=multipart/form-
data>
<input type="file" name="myfile" class="about-btn scrollto btn-
danger" required>
<input type="submit" value="Submit" class="about-btn scrollto
btn-danger" required="true">
</form>
</div>
</section>

<main id="main">

<!--===================== Speakers Section =================== >

<section id="speakers" class="wow fadeInUp">
<div class="container">
<div class="section-header">
<h2>Our Team</h2>
<p>Here are some of our developers</p>
</div>

<div class="row">
<div class="col-lg-4 col-md-6">
<div class="speaker">
<img src="static/img/speakers/1 (3).jpg" alt="Speaker 1"
class="img-fluid">
<div class="details">
<h3><a href="speaker-details.html">RajKumar</a></h3>
<p>Backend Developer</p>
<div class="social">
<a href=""><i class="fa fa-twitter"></i></a>
<a href=""><i class="fa fa-facebook"></i></a>
<a href=""><i class="fa fa-google-plus"></i></a>
<a href=""><i class="fa fa-linkedin"></i></a>
```

```
</div>
</div>
</div>
</div>
<div class="col-lg-4 col-md-6">
<div class="speaker">
<img src="static/img/speakers/2.jpg" alt="Speaker 2" class="img-
fluid">
<div class="details">
<h3><a href="speaker-details.html">Nisha</a></h3>
<p>Frontend Developer</p>
<div class="social">
<a href=""><i class="fa fa-twitter"></i></a>
<a href=""><i class="fa fa-facebook"></i></a>
<a href=""><i class="fa fa-google-plus"></i></a>
<a href=""><i class="fa fa-linkedin"></i></a>
</div>
</div>
</div>
</div>
<div class="col-lg-4 col-md-6">
<div class="speaker">
<img src="static/img/speakers/1 (3).jpg" alt="Speaker 3"
class="img-fluid">
<div class="details">
<h3><a href="speaker-details.html">Rohit</a></h3>
<p>Assistent Developer</p>
<div class="social">
<a href=""><i class="fa fa-twitter"></i></a>
<a href=""><i class="fa fa-facebook"></i></a>
<a href=""><i class="fa fa-google-plus"></i></a>
<a href=""><i class="fa fa-linkedin"></i></a>
</div>
</div>
</div>
</div>
</div>
</div>
</section>

<!--===================    Footer   =========================-->
<footer id="footer">
<div class="footer-top">
<div class="container">
<div class="row">
<div class="col-lg-3 col-md-6 footer-contact">
<h4>Contact Us</h4>
```

```
<p>
Azad Institute of Engineering and Technology <br>
<strong>Phone:</strong>378529365<br>
<strong>Email:</strong> nisha.renusagar@gmail.com<br>
</p>
<div class="social-links">
<a href="#" class="twitter"><i class="fa fa-twitter"></i></a>
<a href="#" class="facebook"><i class="fa fa-facebook"></i></a>
<a href="#" class="instagram"><i class="fa fa-
instagram"></i></a>
<a href="#" class="google-plus"><i class="fa fa-google-
plus"></i></a>
<a href="#" class="linkedin"><i class="fa fa-linkedin"></i></a>
</div>
</div>
</div>
</div>
</div>
</footer>
<a href="#" class="back-to-top"><i class="fa fa-angle-
up"></i></a>

<!=================== JavaScript Libraries ================ >

<script src="{% static 'lib/jquery/jquery.min.js' %}"></script>
<script src="{% static 'lib/jquery/jquery-migrate.min.js'
%}"></script>
<script src="{% static
'lib/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
<script src="{% static 'lib/easing/easing.min.js' %}"></script>
<script src="{% static 'lib/superfish/hoverIntent.js'
%}"></script>
<script src="{% static 'lib/superfish/superfish.min.js'
%}"></script>
<script src="{% static 'lib/wow/wow.min.js' %}"></script>
<script src="{% static 'lib/venobox/venobox.min.js'
%}"></script>
<script src="{% static 'lib/owlcarousel/owl.carousel.min.js'
%}"></script>

<!------- Contact Form JavaScript File ----- >
<! ---- Template Main Javascript File ---->
<script src="{% static 'js/main.js' %}"></script>


<! ======================Camera JavaScript =============== >
```

```
<script type="text/javaScript">

function cam(){
var video=document.querySelector("#videoElement");

navigator.getUserMedia = navigator.getUserMedia ||
navigator.webkitGetUserMedia || navigator.mozGetUserMedia ||
navigator.msGetUserMedia || navigator.oGetUserMedia;

if(navigator.mediaDevices.getUserMedia){
navigator.mediaDevices.getUserMedia({video:true}).then
(function(stream){
video.srcObject = stream
video.play();
}).catch(function (error){
console.log(error);
});
}}
</script>
</body>
</html>
```

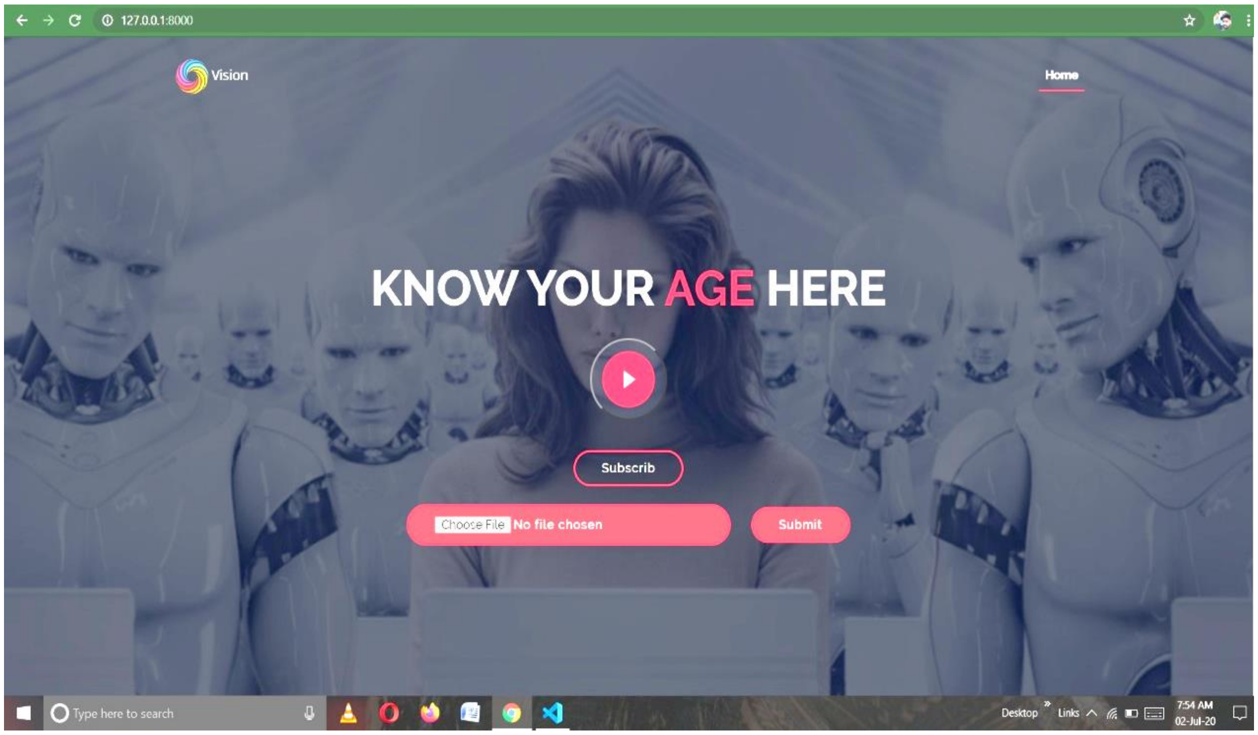


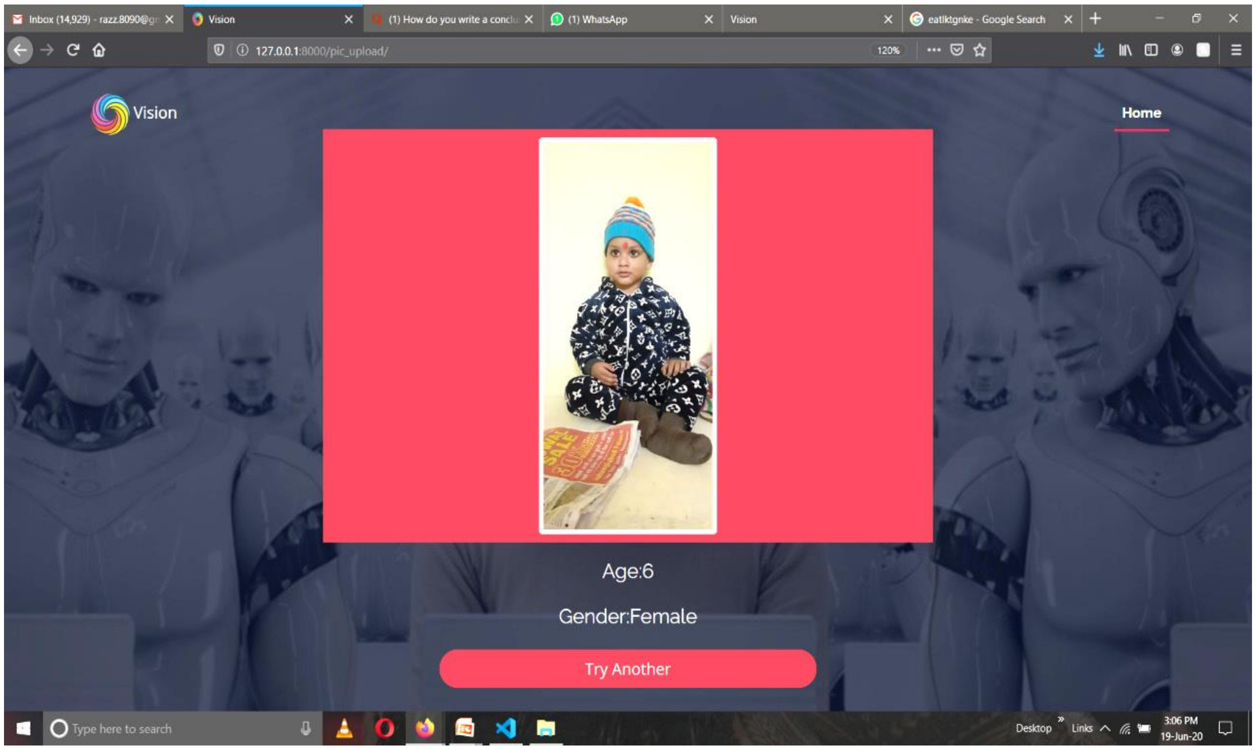
Figure 27: Front view of application

4.3.2 Showing Output

```
{% load static %}
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="utf-8">
<title>Vision</title>
<meta content="width=device-width, initial-scale=1.0"
name="viewport">
<meta content="" name="keywords">
<meta content="" name="description">

<!-- Favicons -->
<link href="{% static 'img/logo (2).png' %}" rel="icon">
<link href="{% static 'img/apple-touch-icon.png' %}" rel="apple-
touch-icon">
<!-- Google Fonts -->
<link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i
,400,400i,700,700i|Raleway:300,400,500,700,800"
rel="stylesheet">

<!-- Bootstrap CSS File -->
<link href="{% static 'lib/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">

<!-- Libraries CSS Files -->
<link href="{% static 'lib/font-awesome/css/font-
awesome.min.css' %}" rel="stylesheet">
<link href="{% static 'lib/animate/animate.min.css' %}"
rel="stylesheet">
<link href="{% static 'lib/venobox/venobox.css' %}"
rel="stylesheet">
<link href="{% static
'lib/owlcarousel/assets/owl.carousel.min.css' %}"
rel="stylesheet">

<!-- Main Stylesheet File -->
<link href="{% static 'css/style.css' %}" rel="stylesheet">

</head>

<body>
```

```
<!--==================== Header========================= >
<header id="header">
<div class="container">

<div id="logo" class="pull-left text-light">
<img src="{% static 'img/logo (2).png' %}" alt=""
title="">Vision
</div>

<nav id="nav-menu-container">
<ul class="nav-menu">
<li class="menu-active"><a href="/">Home</a></li>
</ul>
</nav>
</div>
</header>
<! =================== Output Image Box=================>

<section id="intro">
<div class="intro-container wow fadeIn">
<div class="mb-3 p-2 bg-danger h-75 w-50">
<img class=" h-100 img-thumbnail img-fluid" src="{{pic_name}}">
</div>

<div class="col-md-4">
<h5 class="text-light">Age:{{age}}</h5>
<h5 class="text-light">Gender:{{gender}}</h5>
<h5 class="text-light">{{error}}</h5>
<button class="btn btn-block btn-danger badge-pill"><a href="/"
class="text-light">Try Another</button>

</div>
</div>
</section>
< ===================== Script Files ================>

<script src="{% static 'lib/jquery/jquery.min.js' %}"></script>
<script src="{% static 'lib/jquery/jquery-migrate.min.js'
%}"></script>

<script src="{% static
'lib/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
<script src="{% static 'lib/easing/easing.min.js' %}"></script>
<script src="{% static 'lib/superfish/hoverIntent.js'
%}"></script>
```

```
<script src="{% static 'lib/superfish/superfish.min.js'
%}"></script>
<script src="{% static 'lib/wow/wow.min.js' %}"></script>
<script src="{% static 'lib/venobox/venobox.min.js'
%}"></script>
<script src="{% static 'lib/owlcarousel/owl.carousel.min.js'
%}"></script>
<!-- Contact Form JavaScript File -->
<script src="{% static 'contactform/contactform.js'
%}"></script>
<!-- Template Main Javascript File -->
<script src="{% static 'js/main.js' %}"></script>
</body>
</html>
```



Age:6

Gender:Female

Figure 28: Page showing output of the prediction

4.3.3 Subscription Form

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Vision</title>
<meta content="width=device-width, initial-scale=1.0"
name="viewport">
<meta content="" name="keywords">
<meta content="" name="description">
<!-- Favicons -->
<link href="{% static 'img/logo (2).png' %}" rel="icon">
<link href="{% static 'img/apple-touch-icon.png' %}" rel="apple-
touch-icon">
< ==================================== CDN
=============================>
<!-- Google Fonts -->
<linkhref="https://fonts.googleapis.com/css?family=Open+Sans:300
,300i,400,400i,700,700i|Raleway:300,400,500,700,800"
rel="stylesheet">
<!-- Bootstrap CSS File -->
<link href="{% static 'lib/bootstrap/css/bootstrap.min.css' %}"
rel="stylesheet">
<!-- Libraries CSS Files -->
<link href="{% static 'lib/font-awesome/css/font-
awesome.min.css' %}" rel="stylesheet">
<link href="{% static 'lib/animate/animate.min.css' %}"
rel="stylesheet">
<link href="{% static 'lib/venobox/venobox.css' %}"
rel="stylesheet">
<link href="{% static
'lib/owlcarousel/assets/owl.carousel.min.css' %}"
rel="stylesheet">
<!-- Main Stylesheet File -->
<link href="{% static 'css/style.css' %}" rel="stylesheet">
</head>

<body>

<!--==========================    Header
==============================-->
<header id="header">
<div class="container">
<div id="logo" class="pull-left text-light">
```

```
<!-- Uncomment below if you prefer to use a text logo -->
<!-- <h1><a href="#main">C<span>o</span>nf</a></h1>-->
<img src="{% static 'img/logo (2).png' %}" alt=""
title="">Vision
</div>
<nav id="nav-menu-container">
<ul class="nav-menu">
<li class="menu-active"><a href="/">Home</a></li>
</ul>
</nav><!-- #nav-menu-container -->
</div>
</header>
< ================================    Form
==========================>
<section id="intro">
<div class="intro-container wow fadeIn">
<div class="container">
<div class="row my-5 ">
<div class="col-lg-6 mx-auto">
<div class="card mt-5 shadow">
<div class="card-body">
<h4 class="card-title text-center text-danger">Subscribe!</h4>

<form method="POST" action="submit/">
<div class="form-group">
<label for="">First Name</label>
<input type="text" class="form-control" name="name" id="" aria-
describedby="emailHelpId"
placeholder="enter username"></div>
<label for="">Last Name</label>
<input type="text" class="form-control" name="last_name" id=""
aria-describedby="emailHelpId"
placeholder="enter username">
<label for="">Email</label>
<input type="email" class="form-control" name="email" id=""
aria-describedby="emailHelpId"
placeholder="enter username">
<button class="btn btn-danger btn-block"
role="button">Subscribe</button>
</form>

</div>
</div>
</div>
</div>
</div>
</div>
```

```
</section>

< ============================== Script Files
======================== >

<script src="{% static 'lib/jquery/jquery.min.js' %}"></script>
<script src="{% static 'lib/jquery/jquery-migrate.min.js'
%}"></script>
<script src="{% static
'lib/bootstrap/js/bootstrap.bundle.min.js' %}"></script>
<script src="{% static 'lib/easing/easing.min.js' %}"></script>
<script src="{% static 'lib/superfish/hoverIntent.js'
%}"></script>
<script src="{% static 'lib/superfish/superfish.min.js'
%}"></script>
<script src="{% static 'lib/wow/wow.min.js' %}"></script>
<script src="{% static 'lib/venobox/venobox.min.js'
%}"></script>
<script src="{% static 'lib/owlcarousel/owl.carousel.min.js'
%}"></script>
<script src="{% static 'js/main.js' %}"></script>
</body>
</html>
```

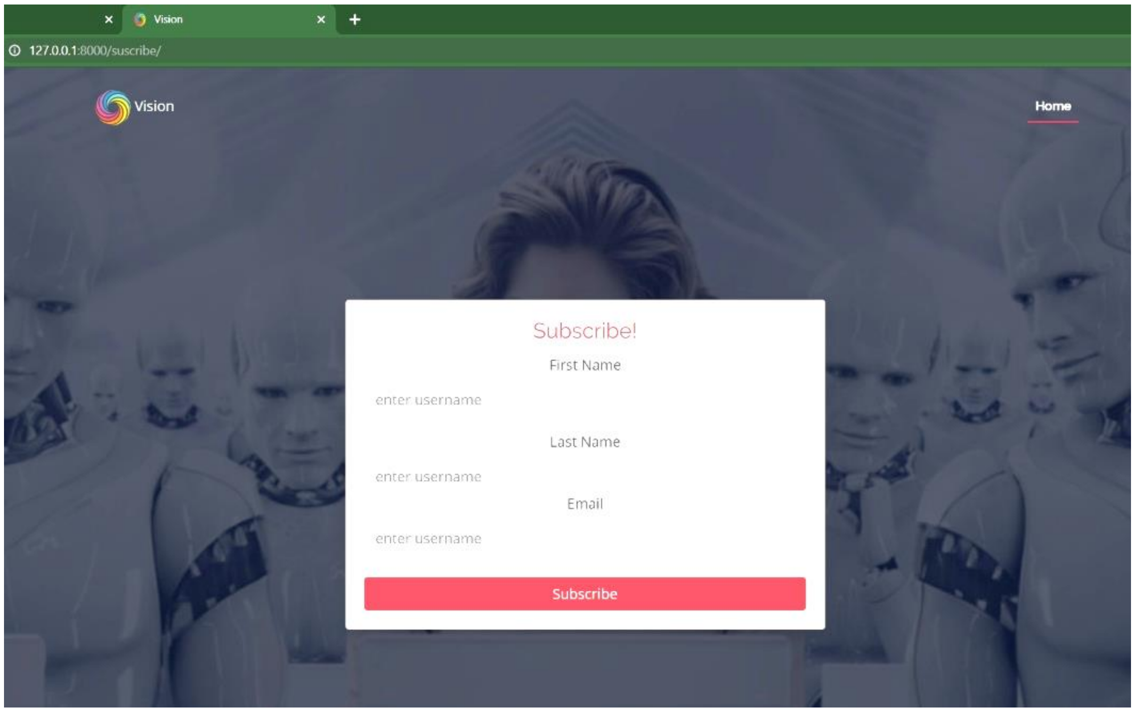
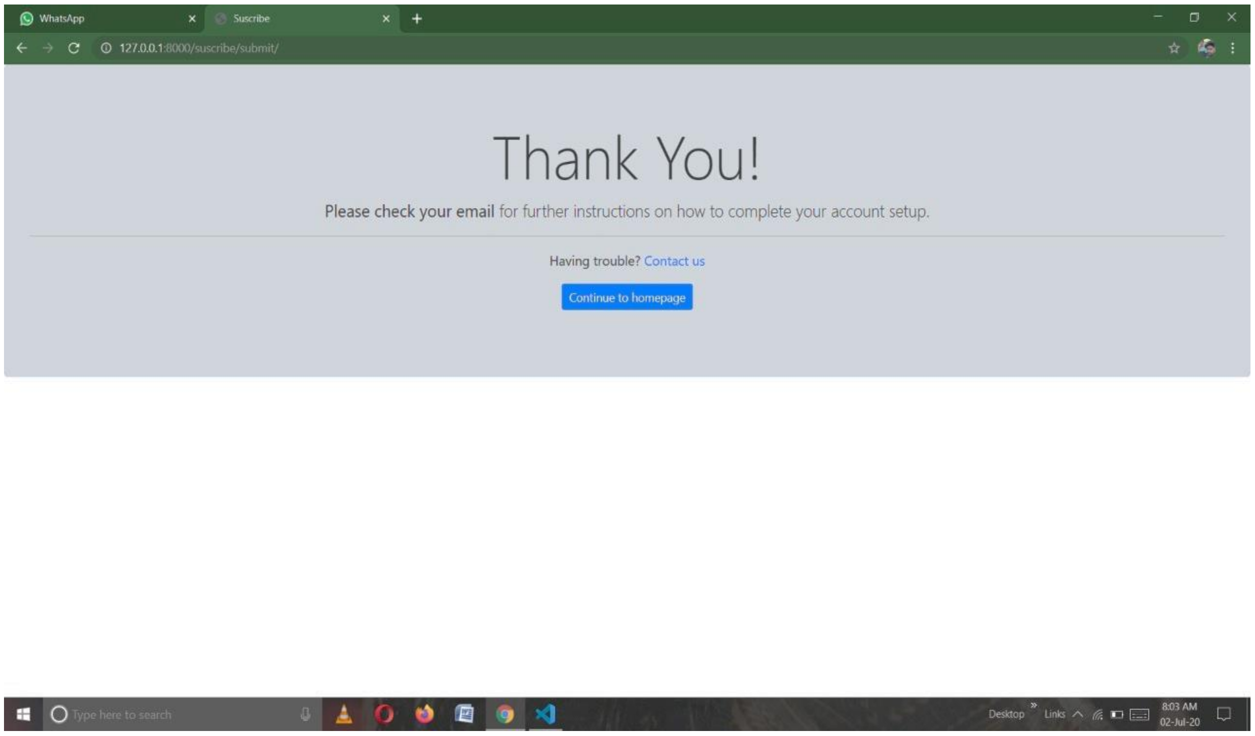


Figure 29: Subscription form for user

4.3.4 Jumbotron

```
{% load static %}
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>Suscribe</title>

< ==================== CDN =====================>

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/boo
tstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T
"crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo
"crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd
/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1
"crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/boots
trap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM
"crossorigin="anonymous"></script>

</head>

<body>
```

```
<div class="jumbotron text-center">
<h1 class="display-3">Thank You!</h1>
<p class="lead"><strong>Please check your email</strong> for
further instructions on how to complete your account setup.</p>
<hr>
<p>
Having trouble? <a href="">Contact us</a>
</p>
<p class="lead">
<a class="btn btn-primary btn-sm" href="/"
role="button">Continue to homepage</a>
</p>
</div>
</body>
</html>
```



Thank You!

Please check your email for further instructions on how to complete your account setup.

Having trouble? Contact us

Figure 30: After subscription successful

5. CONCLUSION

This paper introduced an approach to classify facial images into their corresponding gender and age. The main emphasis of this research is to apply the training and learning process of the human brain in pattern recognition and classification from the normal computer. Automatic classification of facial images into age and gender has been used in several applications in the commercial world such as video surveillance systems and enhance image searching in search engines. The proposed methodology used parameters taken from the geometric facial feature variations influenced by the two gender types and the facial skin texture variation in the ageing process. These parameters are then used to classify the images into corresponding gender and age by using a neural network. Facial images for our research are taken from the two databases namely FERET and FGNET which include the gender and age details along with each image.

We use facial images in the range of 8 – 63 as it is very difficult to recognize the gender of little babies by the geometric facial feature variations and also there are not enough facial images of people older than 64 to use in our experiment. For the effectiveness of identifying the age of a given facial image we have used four age groups namely 8 – 13, 14 – 25, 26 – 45 and 45 – 63 where each range has remarkable variations. We have used nearly 550 images for the training process including both male and female images within our interested age groups. Face detection and the facial parts such as eyes, nose and mouth area are located using the OpenCV Haar cascade classifiers. After that feature points required for the experiment are extracted using horizontal and vertical projections. There are many difficulties of correctly locating the feature points automatically in some cases when the face area contains birthmarks like patches and also due to different lighting conditions.

Parameters in the training data set is input to the neural network to train the system and it is used to classify a given query image into the corresponding gender and age group according to the learning done from the parameters used in the training process. There are two neural networks used for the classification of age and gender. Each of the neural networks performs accurately with one hidden layer comprising with suitable number of neurons.

According to the results taken from the proposed age and gender classification methodology, gender classification from facial images has significantly higher accuracy in classification training data as well as testing data. For the testing data the gender classifier results with a correct classification of 85.83% while the training data set gives 89.92% accuracy when using data from 40 images from each data set. This implies that the classification accuracy for the testing data set is very closer to the classification rate for the training dataset. This can be further interpreted that our proposed method for gender classification is a considerably accurate mechanism and it can be used as a successful mechanism in gender classification. According to the results taken from age classifier we can see that the accuracy for the training data set is 79.09% and 74.39% of accuracy for the testing data where we have used data from 40 images. Overall accuracy for the age and gender classifier is 70.5% which is a considerable high accuracy when compared to the overall classification accuracy by human brain which gives a 75% of accuracy. This shows that gender classification by human brain is 100% but when we come to age classification it is 75%, which shows the difficulty for the classification. Therefore we can conclude that the proposed age and gender classification methodology can be accepted as a successful mechanism.

Proposed methodology can be improved further to gain higher accuracy in classification. More parameters can be added to represent the geometric variation of gender to our classifier in order to increase the performance. Reducing the gap between age ranges would result with a better classification of images in to several age groups.

Identification of wrinkles is not 100% accurate as it depends on several external conditions such as facial make-up and smoothing done for the images. Therefore in future it is required to find out another set of parameters which do not cause these problems. Proposed algorithm can be enhanced to perform accurately under any condition of images in the future. Reducing the gap between the present age ranges would also be a reasonable suggestion to end up with a better classification.

5.1 Limitations

- This is not useful for multiple faces at same time.
- People tend to hide their age. Accessories worn could make it worse to detect the correct age.
- The age detected would be an approx. one.

5.2 Future Scope

- The project has a vast scope in present scenario and future.
- It can be further extended with a detection of emotion and can be used in computer vision enabled robots.
- It can be extended to have a detection of mask on the persons face which will help in present scenario of covid-19 Pandemic.
- Useable in different platforms.

5.3 Applications

1. It can be used at places where entry is restricted up to a certain age like movie halls, gym, zoo, pubs, swimming pools, playgrounds etc.
2. It can be used by various beauty camera application providers to predict the age and gender and accordingly provide the suggestions.
3. Some hospitals have a computer based slip issuing system. The application can be beneficial there for allotting the slip and suggesting doctor based on age and gender of the patient.
4. It has a great Scope in security systems both at home or professional level.
5. It can be used in websites, applications and game for age verification process.

6. References

I. https://data.vision.ee.ethz.ch/cvl/rrothe/imdb-wiki/static/wiki.tar.gz

II. https://docs.djangoproject.com/en/3.0/

III. https://docs.djangoproject.com/en/3.0/intro/tutorial02/

IV. https://docs.djangoproject.com/en/3.0/intro/tutorial03/

V. https://docs.djangoproject.com/en/3.0/intro/contributing/

VI. https://en.wikipedia.org/wiki/Convolutional_neural_network

VII. https://cs231n.github.io/convolutional-networks/

VIII. https://heartbeat.fritz.ai/a-research-guide-to-convolution-neural-networks-a4589b6ca9bd

IX. https://matplotlib.org/tutorials/index.html

X. https://pandas.pydata.org/docs/

XI. https://numpy.org/

XII. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

XIII. https://getbootstrap.com/docs/3.3/

XIV. https://towardsdatascience.com/useful-plots-to-diagnose-your-neural-network-521907fa2f45