MA336: Artificial intelligence and machine learning with applications

Final Project

Ahmad Afzaal Cheema
2111168

Introduction

Technological advancements in the world have been proven to be revolutionary since the past couple of decades. With the increasing development in the Information technology world, Artificial intelligence has been taking over and scientists are eager to give machines almost every humanoid functionality. Image recognition/ object recognition is one of the main examples. It was introduced in machines with which these were able to identify the object in front of them. Making machines more humanoid evolved over time bringing face recognition and now gender recognition is under argue.

Latest technology in the world of security of data is face recognition using 'Artificial Intelligence'. Likewise, gender recognition is also important in the upcoming future. It will play a vital role in improving human computer interaction. Ads made for mobiles can be specially categorized based on gender of the person. It can help in recognizing criminals in certain cases or at least reduce the suspects. This feature will also help in population census and counting ratio of men to women.

In this project, we do gender recognition using Convolutional neural network (CNN). Keeping in mind the importance of face recognition in a numerous number of applications and security systems, we are taking face recognition one step further, by adding the 'gender recognition' feature by using CNN.

Analyzing Data

Data Collection

The male and female dataset have been taken from the GitHub database. In total the dataset used in our project is 2307 images. 1173 off 2300 are male photos and for females the numbers are 1134. Figure 1 shows some samples from dataset.

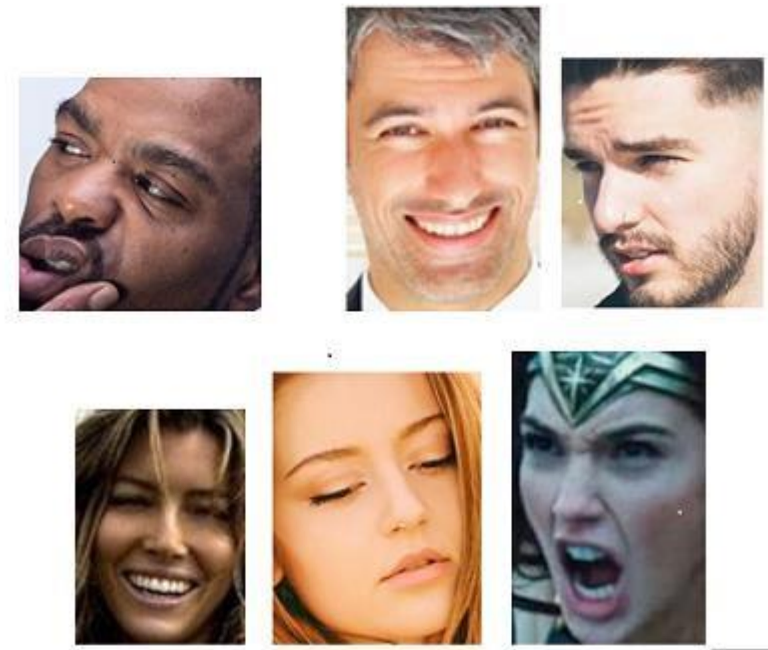

Figure 1: Samples of the dataset

The dataset is organized in 3 folders (train, test, Val) and contains subfolders for each image category (male/female). Table 1 shows information about dataset.

	Quantity	Format	Source
Male	1173	jpg	GitHub
Female	1134	jpg	GitHub
Total	2307	jpg	GitHub

Table 1: Dataset Information

Data Preprocessing

Data preprocessing follows the production of the dataset. The dataset is preprocessed in the manner of training and testing. Preprocessing steps include:

- Resizing
- Shuffling
- Augmentation

Resizing

The first step in preprocessing is resizing the images. CNN takes the input images with same sizes. If the sizes of the images are different it causes difficulty in taking the input and during training. So, the size of the images must be same. The size we used is 96*96 which is often suggested for smaller CNNs.

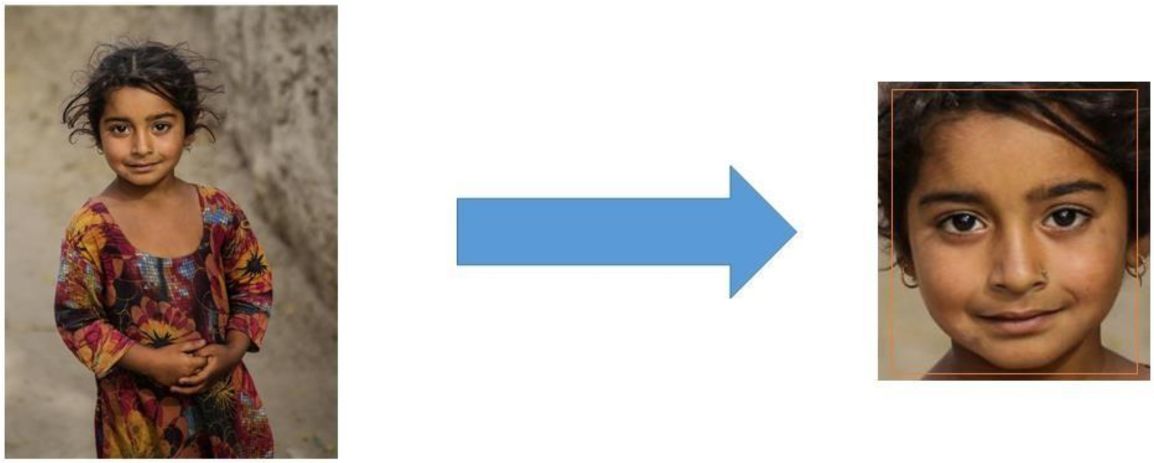

Figure 2 Image Resizing

Shuffling

Shuffling the dataset or images is important in preprocessing. It is a vital to shuffle data for every epoch, because every epoch will have different set of images/batch. Shuffling the data also results in better training of the system and helps increase the accuracy of the system.

Data Augmentation

In order to increase the dataset and enhance it for better training we applied data augmentation technique. Table 2 shows details of augmentation parameters

parameters	Range values
Rotation range	25
Width shift range	0.1
Height shift range	0.1
Shear range	0.2
Zoom range	0.2

Table 2: Augmentation parameters

Methodology

CNN (Convolutional Neural Network)

Convolutional Neural Networks is 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 CNNs to learn the important objects present in the image, allowing them to discern one image from the other. In figure 3 shown below, the CNN has detected a Face and recognized it as a male.

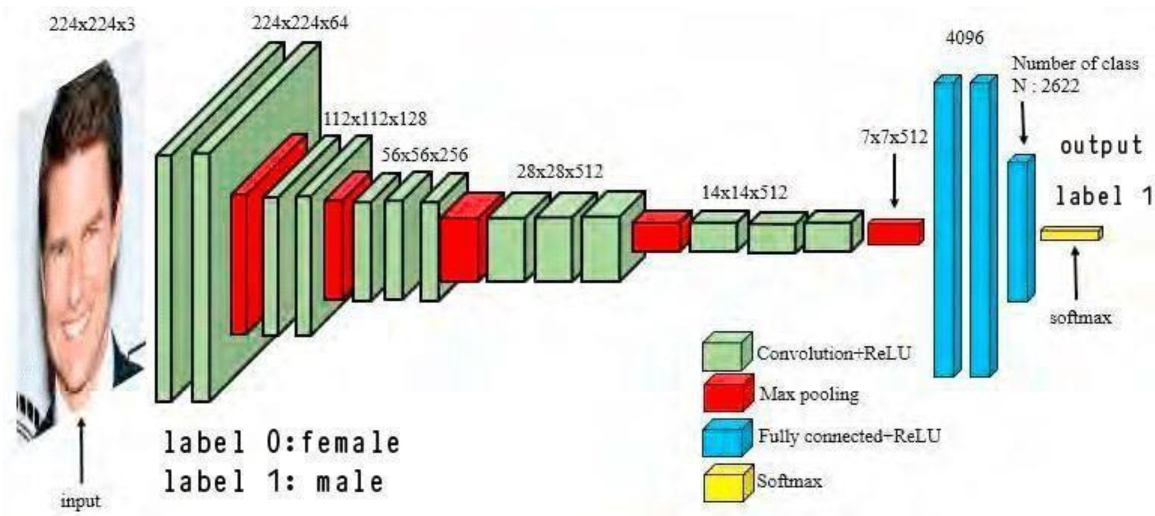

Figure 3: Example of CNN

A CNN has three layers input layer, hidden layer, and an output layer. The input layer is a simple passive node that only transfers a single value/image to the hidden layers. The hidden and output layers are the active nodes. The hidden layers then convolve the input image or value and send the results to the next layer. The output layers simply show the results. The layers of a CNN include convolution, max pooling, flatten and dense layers. They can either be 3D or 2D. RGB or colored images are 3-dimensional. In figure 4 architecture of CNN & its layers are briefly explained.

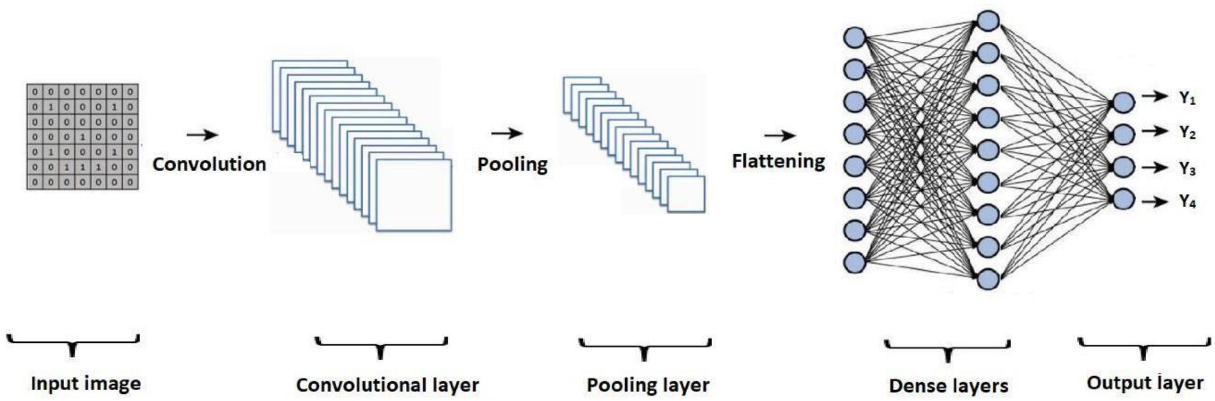

Figure 4 Representation of the architecture of a CNN

In this CNN architecture there are different steps to train the model. The steps include:

- **Input Layer:** Input layer has nodes according to the number of inputs. This system has inputs labelled either as man or woman.
- **Convolutional Layers:** The convolutional layers convolve the input and apply filters on it. In this model 5 convolution layers have been used for training of the model.
- **Pooling Layer:** There are different options of pooling like max pooling, average pooling etc. in this project we use max pooling. This step manipulates with high pixel points of the convolving image. The input image is refined after every pooling layer.
- **Dropout:** When pooling layer is finished, all the noisy data in the input is dropped out by the dropout function. 0.25% of the convolving image's data is dropped out after pooling layer in this case.
- **Dense Layers/Fully connected layers:** All the pre-processed data from previous convolution layers are passed to the dense layers. Dense layers simply predict the probability of the convolving image based on fully connected neurons.
- **Output Layer:** The output layer displays the result of the input with a probability of it being a male's picture or a female's picture.

Feature Extraction

Input images in the dataset are converted into 2D arrays. In feature extraction input image is passed through a 2D convolutional network which consists of 32 filters. The size of each filter is 3*3. After convolutional max pooling and dropout is applied. After this, 2 more 2D convolutional layers consisting of 64 filters each having the filter size 3*3. Same process is repeated with last 2 more 2D convolutional layers consisting of 128 filters each having the filter size 3*3. Then we do flattening. Flatten function converts a 2-Dimensional image into 1-Dimensional image. Figure 5 shows the feature extraction process of our project.

Figure 5 Feature Extraction

Classification:

Once the feature extraction is done the next step comes is classification. In the output layer we have two nodes one for male and other for female. In the output layer we use the 'sigmoid' activation function, it give us better results after hit and trials. Now the CNN classify the input image whether male or female.

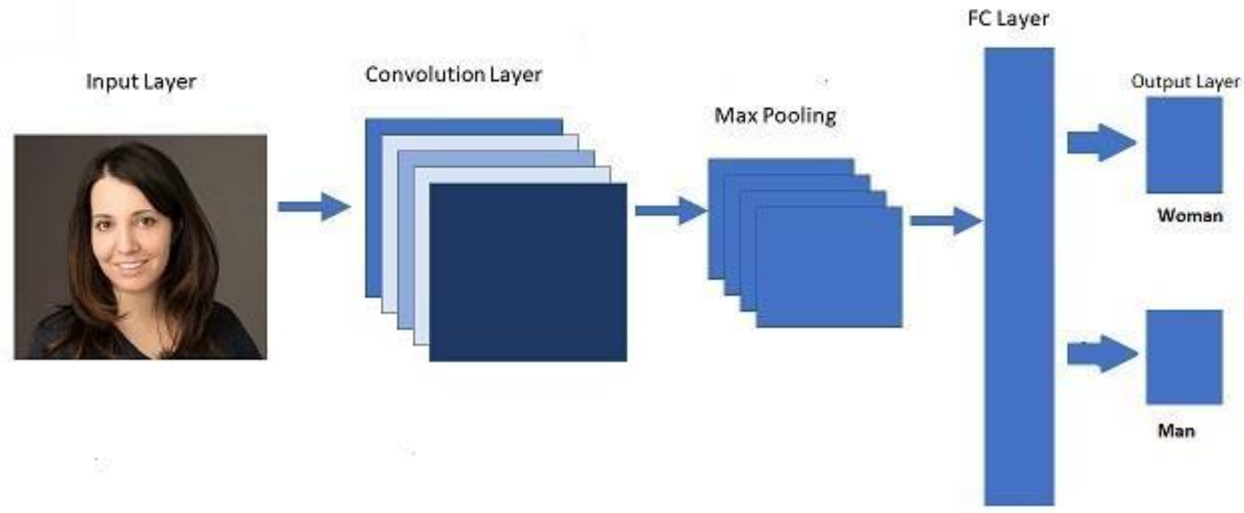

Figure 6: Image Classification using CNN

Results and Discussion

Our objective of the undertaking this is to classify whether the analyzed person is man or woman. For this purpose, the whole dataset was split into two parts as training and test data. The dataset was separated in two folders labeled as man and woman. We trained our system on this organized data. The images for training are 1845. In our model trainable parameters are 8,672,322 and non-trainable forms are 2,880. We use architecture of CNN for feature extraction as well as for classification through its last layer. The model summary shows that which models and layers are present, output shape and prams, no. of trainable parameters and no. of non-trainable parameters show in the figure 7.

```
 dense (Dense)                (None, 1024)              8389632

 activation_5 (Activation)    (None, 1024)              0

 batch_normalization_5 (Batc  (None, 1024)              4096
 hNormalization)

 dropout_3 (Dropout)          (None, 1024)              0

 dense_1 (Dense)              (None, 2)                 2050

 activation_6 (Activation)    (None, 2)                 0

=================================================================
Total params: 8,675,202
Trainable params: 8,672,322
Non-trainable params: 2,880
```

Figure 7: Model Summary

The training accuracy is 96.63%, which shows that are our model is correctly trained.

After completion of model's training, we evaluate our model and we see that outcome is effectively distinguishing between images of man and woman and achieved accuracy 97.19% and cross entropy loss is 0.07. Graphs below in figure 8 and figure 9 shows classification accuracy and cross entropy loss of the model.

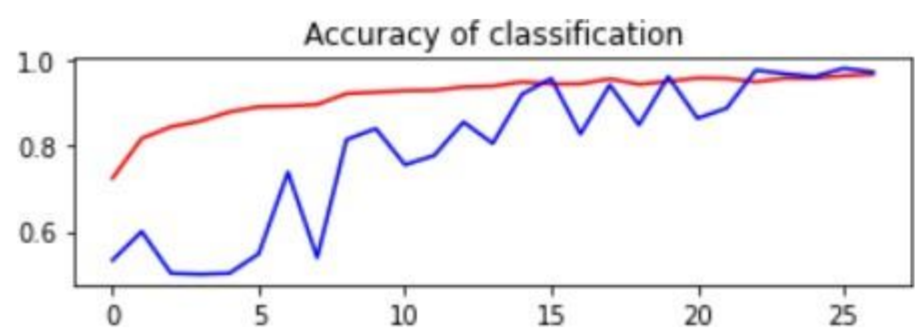

Figure 8: Classification Accuracy of Model

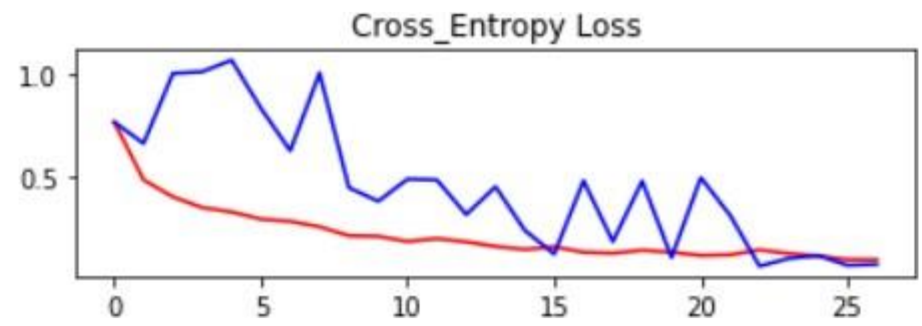

Figure 8: Cross Entropy Loss of Model

Figure 10 shows the confusion matrix for the binary classification of gender recognition.

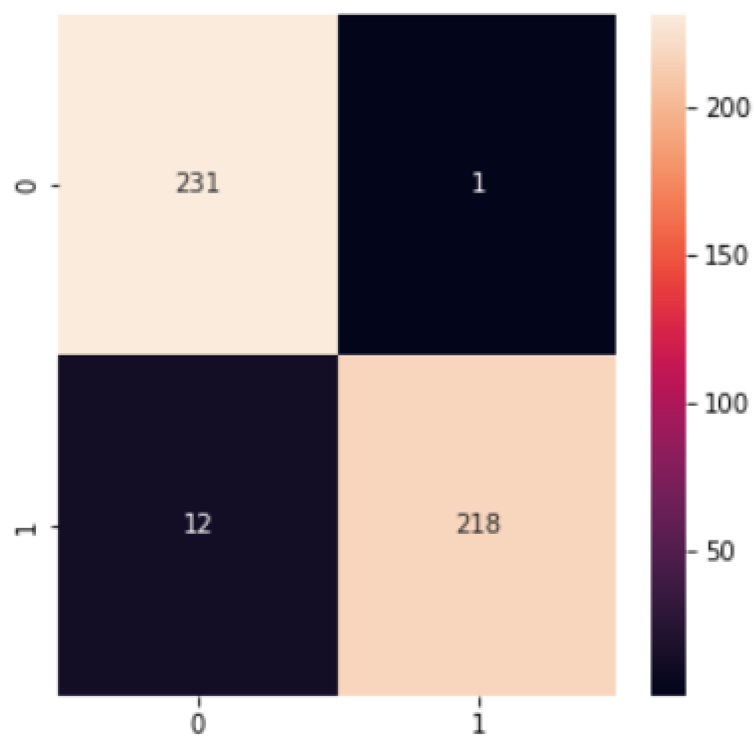

Figure 8: Confusion metrics

Training and Validation Losses and Accuracies

We also calculate the training and validation loss as shown in figure 9.

Figure 9 Training and Validation Loss and Accuracy

Training Loss

The amount of data which is lost during the training phase of the model is termed as training loss. The graph below shows the percentage of the training data loss. When training started the percentage of data loss was 0.7 meaning the data loss in the first 5 epochs was greater. The reason for training loss to be this much was that the model was never trained on this dataset before. After 5 epochs the training loss gradually start decreasing from 0.7 to 0.1 approximately.

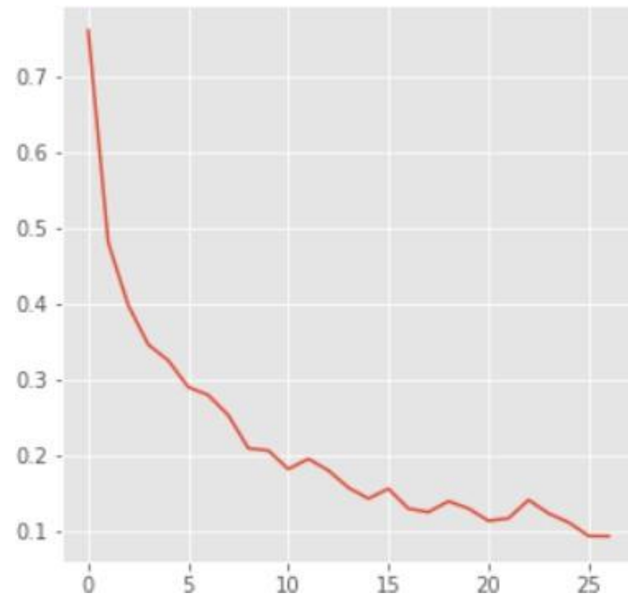

Figure 10: Training Loss

6.2.1 Training Accuracy

Training accuracy means how well the model is trained. In the first five epochs the training accuracy is less than 90%. After this the accuracy of training started rising to the point where the system achieved the training accuracy of almost 1.

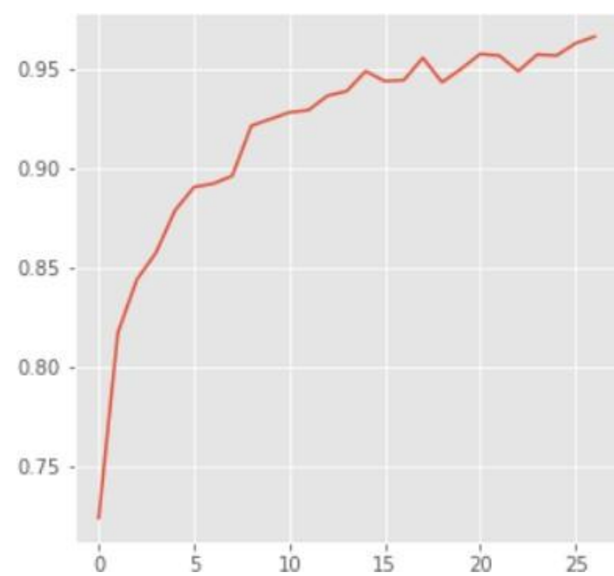

Figure 11: Training Accuracy

6.2.2 Validation Loss

Validation Loss indicates about the model that how well it behaves after each epoch. After each epoch the validation loss changes. Sometimes the loss is greater other times it is lesser. Once training is over the validation loss ultimately goes around 0.

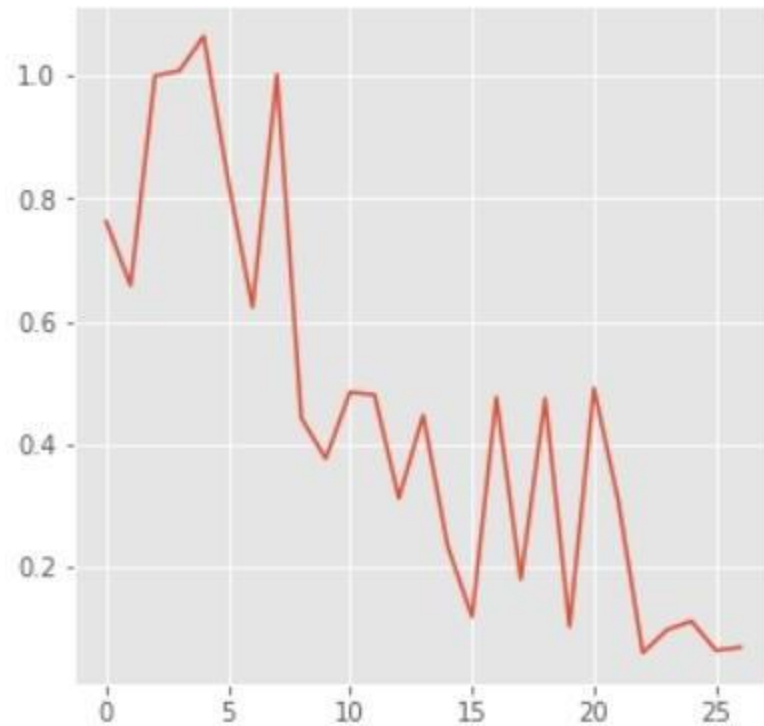

Figure 12 Validation Loss

6.2.3 Validation Accuracy

Validation accuracy can be termed as testing accuracy because it is gained during the training of the model. The dataset which is not used in training phase of the model but used during the training process for validation of the training. In first 5 epochs the validation accuracy is quite low. After 5 epochs the validation of the training gradually started to become stable with small jumps up and down. Final validation accuracy of the model when training ends is 97.19%

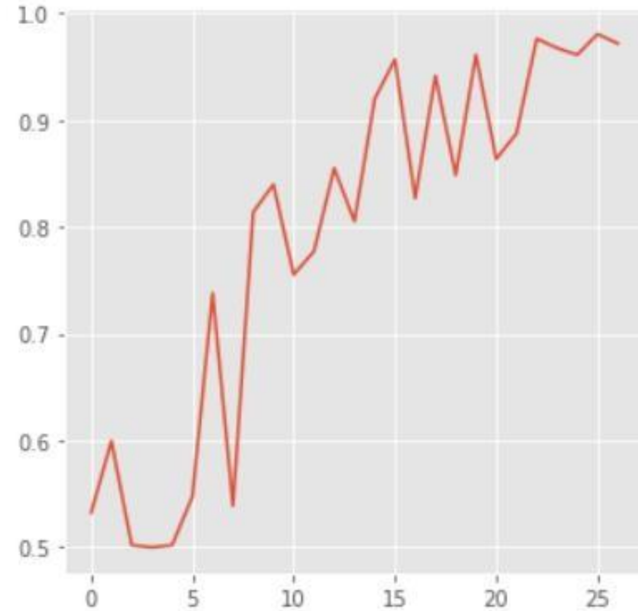

Figure 13: Validation Accuracy

Real Time Testing:

After evaluating the model, we test the model by external images. For this we print a code through which we open webcam and our model classify the image as man or woman. Below there are some examples of this test

Testing Results on Live Camera with Accuracy

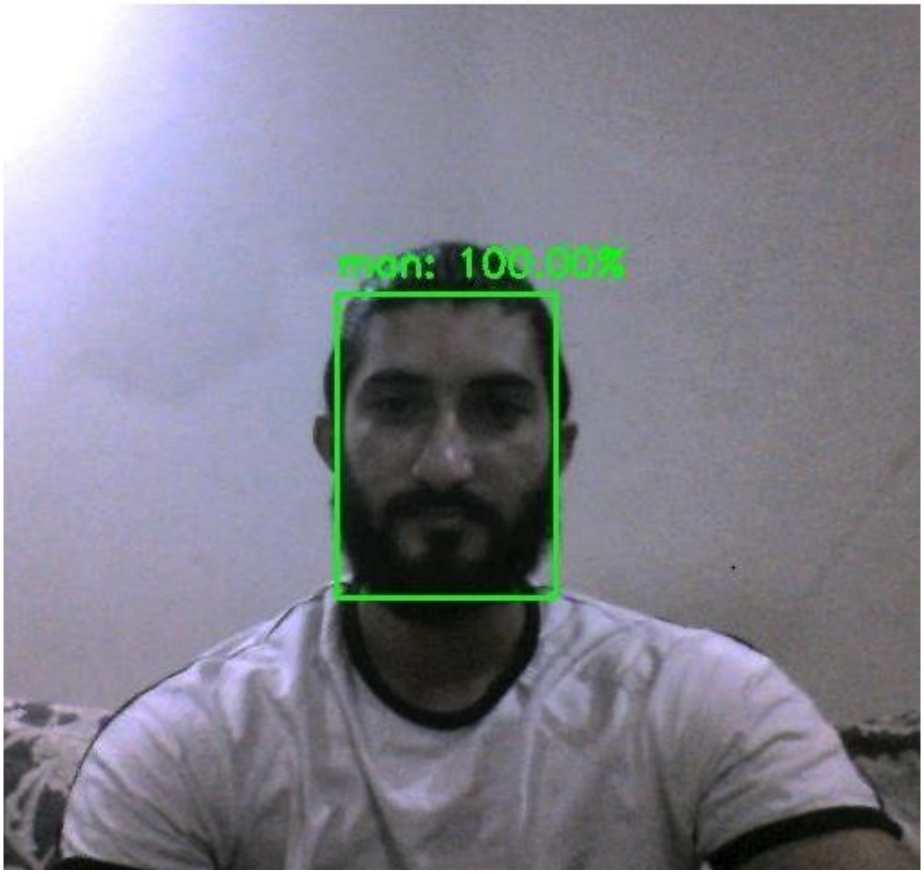

Figure 14 Test Result 1

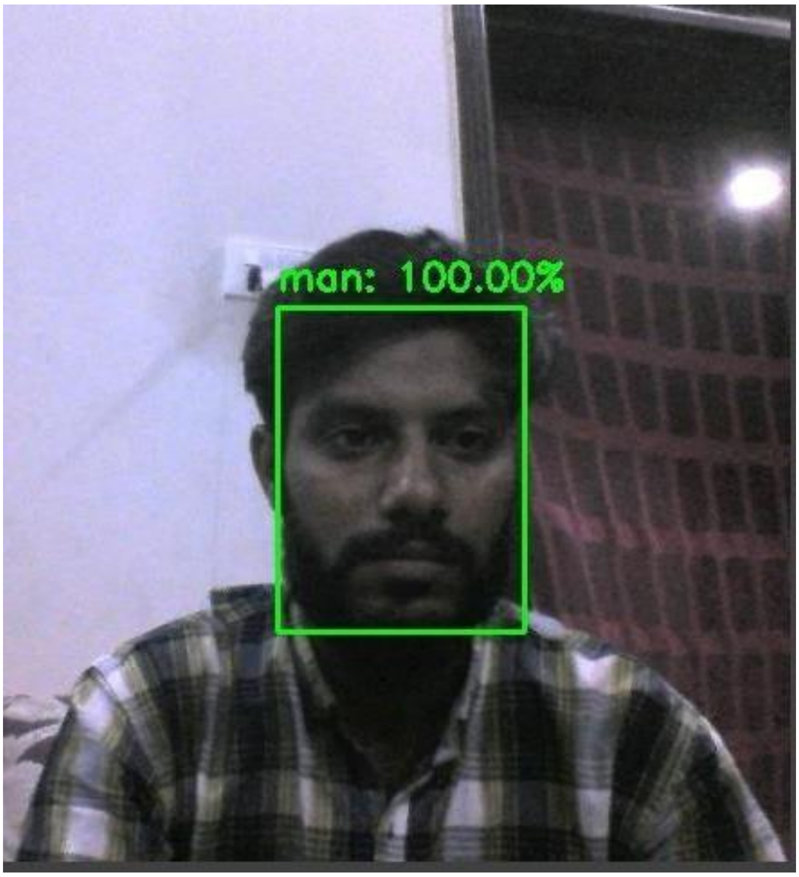

Figure 15: Test Result 2

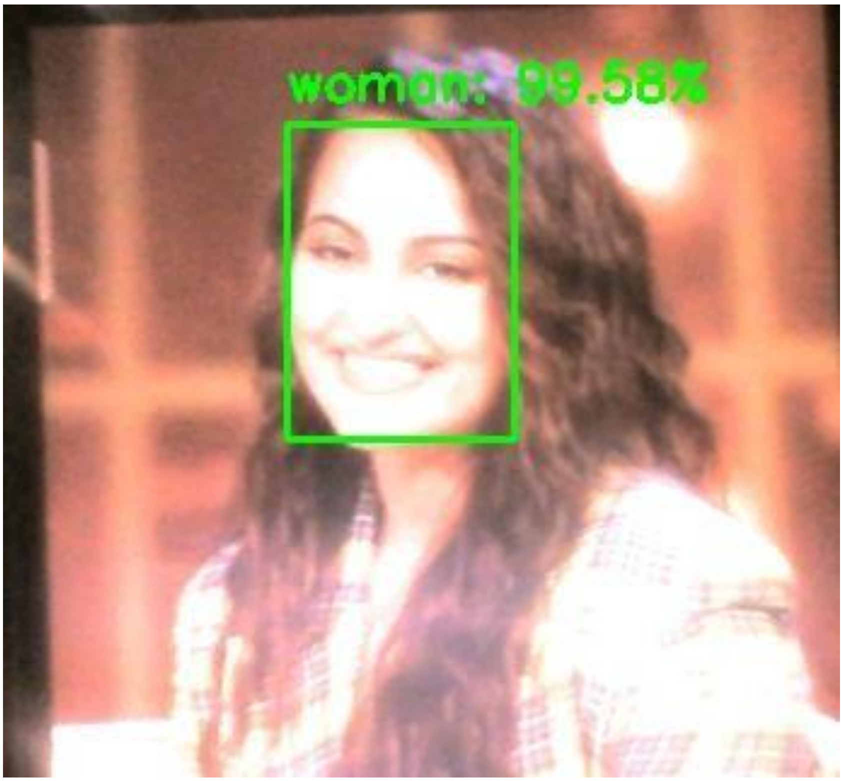

Figure 16: Test Result 3

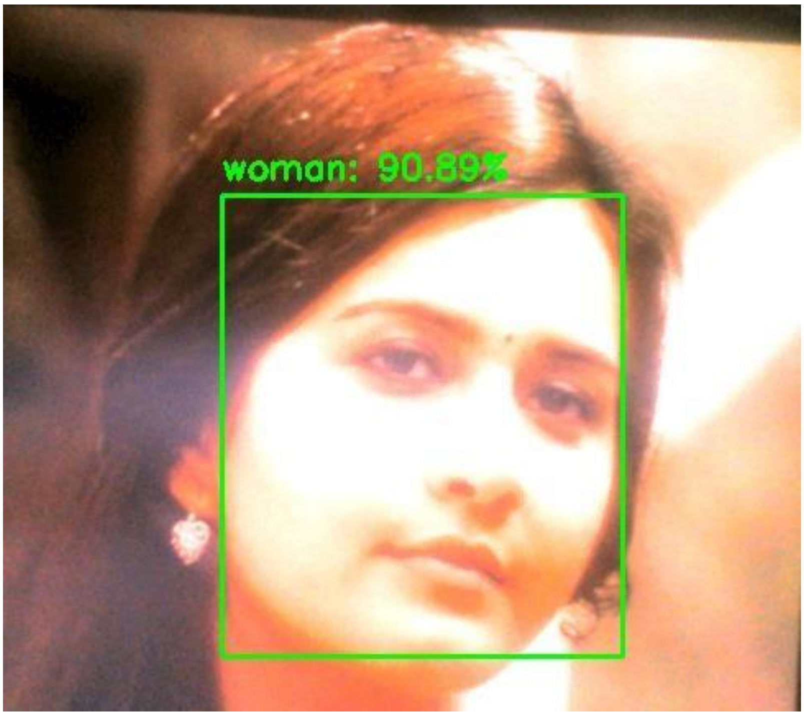

Figure 17: Test Result 4

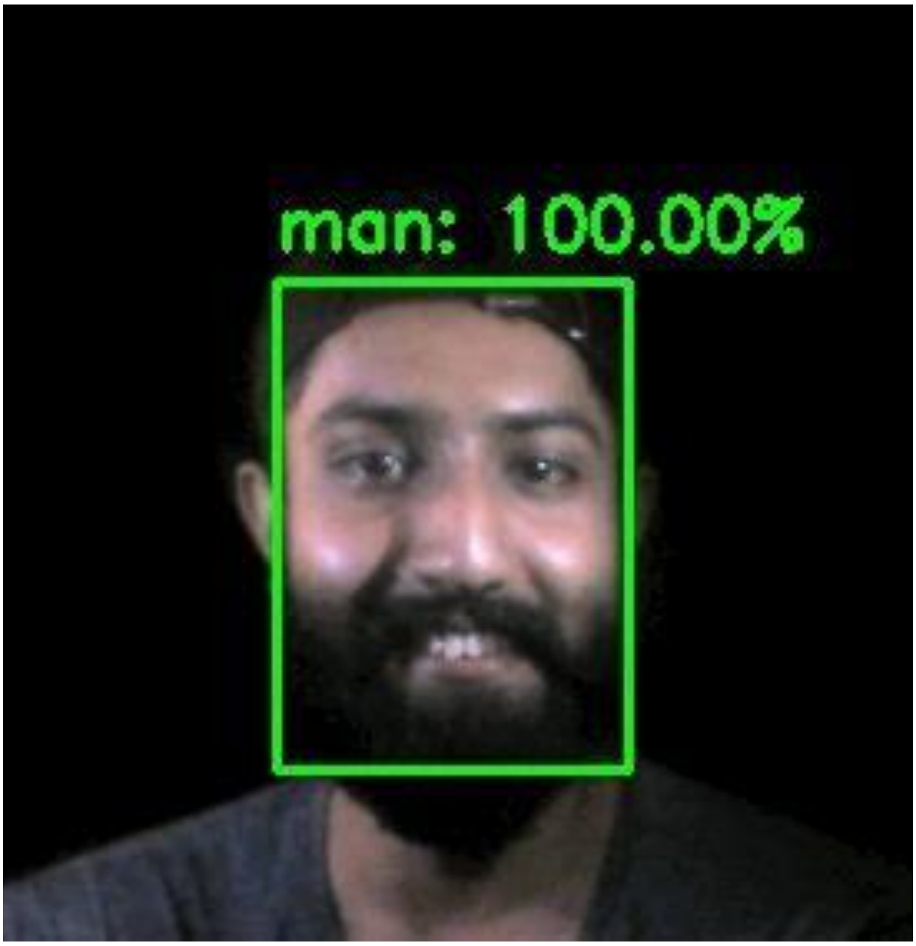

Figure 18: Test Result 5

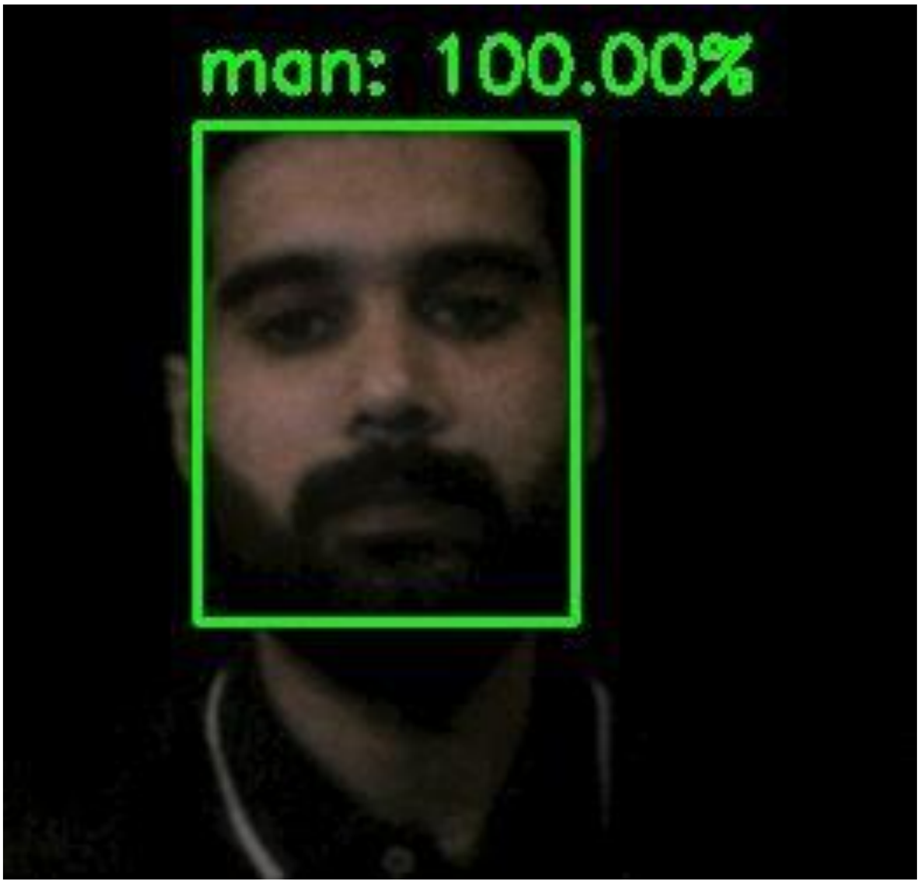

Figure 19: Test Result 6

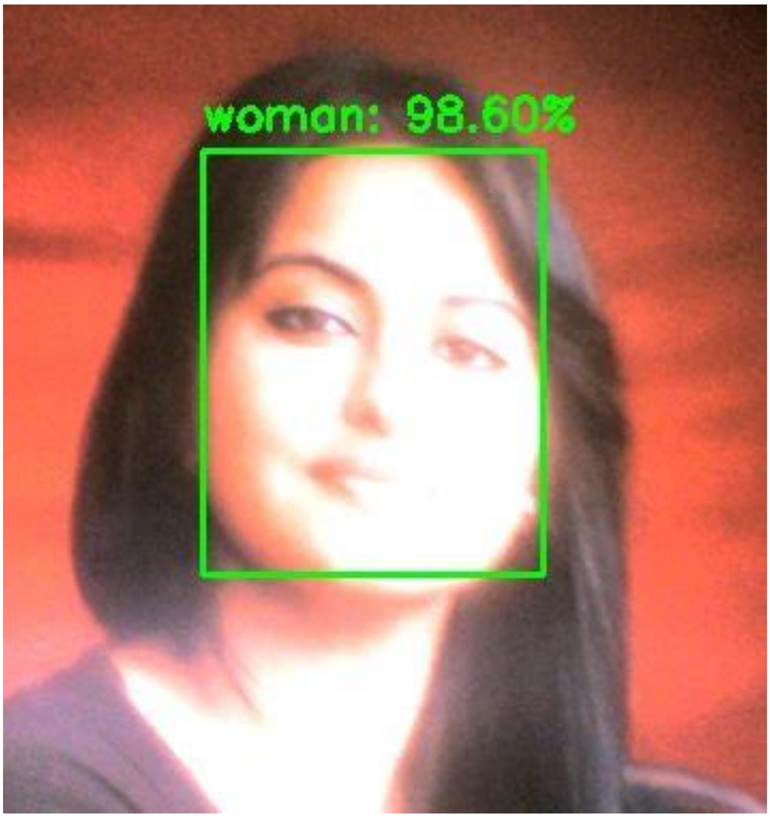

Figure 20: Test Result 7

Conclusion

There were two models in this system. The model that trains and tests the system's predictability ratio. A dataset of 2307 images was given as an input to this model. This dataset was divided into two parts: training dataset and testing dataset. The training dataset was used to train the model. In the training phase the system was trained with the allotted dataset of man and woman labelled images. These images convolve in different convolution layers and then hidden and dense layers. All these layers have different roles. The most important is to predict the decision which is done in the dense layers which are fully connected to each other. In the testing phase, the system was evaluated with the dataset allotted for testing. We use 'accuracy' as performance matric and achieved 97.19% accuracy. We also calculate cross entropy loss, i.e., 0.07, as well as training accuracy and loss. After the completion of training and testing phases this model was then embedded into the second system. The second model captures a face from live video or a photo and recognizes its gender. This model can detect the faces from live videos i.e., CCTV's and other recorded videos, and can be proved useful in many scenarios like recognizing criminals in certain cases or at least reduce the suspects. This feature will also help in population census and counting ratio of men to women.