University of Waterloo

ME 780: Computational Intelligence

Project Report

Analysis of emotions using Hand gestures and facial expressions

Submitted to: Prof. William W. Melek

Name: Hardik Gossain

Roll No.: 20957816

Email: hgossain@uwaterloo.ca

Executive Summary

This project aimed to apply machine learning to applications related to social robotics or we can say Human-Robot Interaction. To do so, mainly recognition of facial expressions and hand gestures was targeted. Things like voice recognition, haptic feedback, and mood analysis using pitch of voice could also be done but for now, the above mentions features were focused on. Apart from HRI, facial expressions recognition has a vast area of application.

In this project, two convolutional neural networks were trained. One was used to determine the facial expressions, the expressions included happy, sad, neutral, surprised, and angry. The other model trained was to recognize hand gestures. The hand gestures that this model could recognize were stopped, okay, thumbs up, and peace. The data set used for training facial expressions recognition was the FER-2013 dataset taken from Kaggle. Similarly, the dataset for hand gesture recognition was also taken from Kaggle. The process of the algorithm made started by training both the models and storing them in a Hierarchical Data Formats (.h5) file. The facial expression recognition model consisted of five convolutional layers. The hand gesture recognition model consisted of three convolutional layers.

Once the model was trained, it used live video feeds to detect facial expressions and hand gestures. For the facial expressions, OpenCV and its haarcascade frontal face classifier were used to detect the face. Frames from the live video were taken and the face in the whole image was detected. The detected face was converted into grayscale and was given as the input to the neural network. The neural network gave a list of probabilities for all the above-mentioned expressions. The expression with the maximum probability was taken as the result and overlayed on the frame as the output. The same process was followed for hand gestures too. The main difference was in the algorithm for detecting the hand and sending it as the input to the other trained model. There was some inefficiency while live detection which is discussed later in this report.

Introduction

The advancement of technology has made many innovative features in the field of robotics possible. This is due to many factors like higher processing power, better sensors, and most importantly, accessibility to these resources. By this, Human-robot interaction (HRI) has got a great advantage. With such advancements, better algorithms are being made so that to some extent robots could understand humans and interact with them. There are many ways by which robots can understand what humans want to convey [1]. It includes voice recognition, facial recognition, etc. In this project, we will be focussing on facial expressions recognition and hand gesture recognition.

Facial emotions are critical elements in human verbal exchange that help to recognize the intentions of others. In general, humans infer the emotions of other human beings, including happiness, sadness, fear, neutrality, and anger, through the usage of facial expressions and vocal tones. Facial expressions are one of the major fact's channels in interpersonal verbal exchange. Therefore, it's far herbal that facial emotion studies have won lots of attention over the past decade with packages in perceptual and cognitive sciences [2]. Interest in automated Facial Emotion Recognition has also been increasing lately with the fast development of Artificial Intelligent (AI) techniques. They are used in many programs and their exposure to human beings is increasing. To enhance Human-Robot Interaction (HRI) and make it more realistic, robots must be provided with the capability to understand the encircling surroundings, especially the intentions of people.

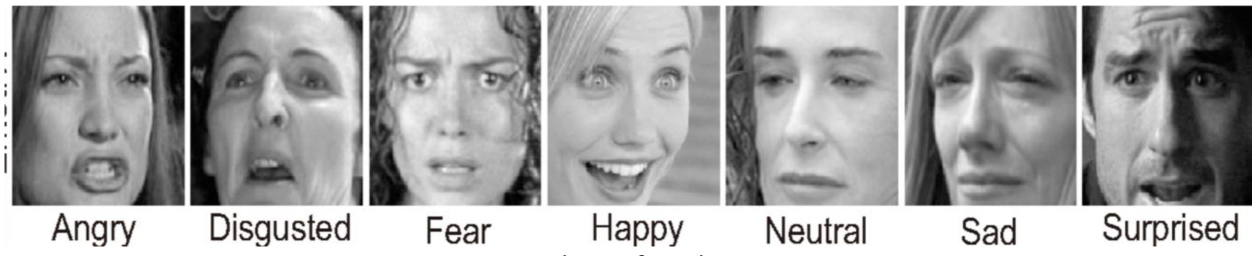


Fig 1: Seven basic facial expressions

Hand-gesture plays an important role in many aspects of human communication. Some people tend to display their confidence and express themselves using them [3]. Most importantly for specially challenged people who cannot speak, sign language is their major means of communication. In places like underwater, where humans cannot communicate verbally, hand signs are the only means of communication for them. The ability to recognize hand gestures, given to robots could be used in many useful ways. In the case of Human-Robot interactions, hand gestures could be used to command and express any feelings to the robot. There are many applications like autonomous drones following hand gestures to perform different functions. Even many gaming consoles like play station combines virtual reality with hand gesture recognition for entertainment purposes.

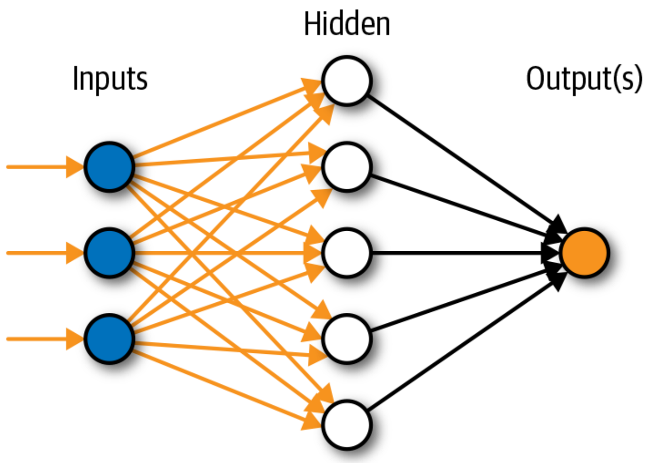


Fig 2: Basic layout of Neural Networks

Machine Learning (ML) has been validated to be one of the maximum recreation-changing technological improvements of the past decade. In the increasingly aggressive corporate global, ML is allowing corporations to fast-song digital transformation and move into an age of automation. Some may even argue that AI/ML is required to stay relevant in some verticals, including virtual payments and fraud detection in banking or product hints [4]. The eventual adoption of device gaining knowledge of algorithms and its pervasiveness in corporations is also properly documented, with specific companies adopting systems getting to know at scale throughout verticals. Today, each different app and software program all around the Internet makes use of machines to get to know in some form or the other. Facial and hand gesture recognition is one of the many applications of Machine Learning. It used similarities in patterns to detect different types of gestures and facial expressions. In this paper, the author has used patterns to identify faces [5]. Similarly, in this paper [6], the authors have detected patterns using CNN for hand gesture classification. Facial expressions and hand gesture recognition play a very important role in social robotics.

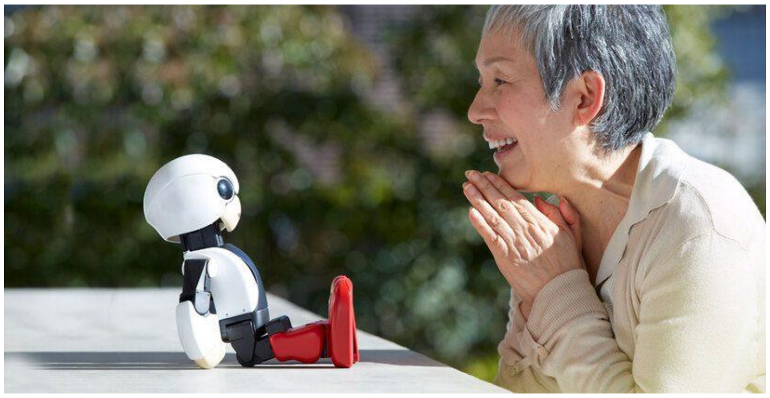


Fig 3: Human-Robot Interaction (HRI)

A social robot is an artificially intelligent (AI) system this is designed to interact with people and different robots. In the workplace, social robots have the capability to take over complete job functions, consisting of greeting and primary customer service. In the house, social robots could come to be useful enough to serve as family members and be purposely designed with precise personalities and quirks to interact own family participants. In this research article [7], facial expressions were studied in a Human-robot interaction.

This project has been inspired by all this research and aimed to develop two neural network models that can recognize facial expressions and hand gestures from a live video feed.

Literature Review

In almost every industry, from artificial intelligence and gaming to marketing and healthcare, facial expression detection is an important topic. [8] Due to their many filters, CNNs perform better for image identification tasks since they can capture the inputs' spatial properties. [8] Six convolutional layers, two max-pooling layers, and two fully connected layers make up the suggested model [8]. To demonstrate the flexibility of the CNN model the facial recognition algorithm is implemented on raspberry pi [9]. To provide identification of a person, a camera will be installed on the side of our glasses and a Raspberry Pi will be kept in the user's pocket. The glasses are made to do facial recognition by collecting data using a camera that is attached to them, then showing the results them [9]. Images are first captured by the camera, then relayed to the CPU, which compares them to databases stored on Storage Devices (SD) and displays the findings on Organic LEDs (O-LED). Not just the face, using the CNN model we can also detect the hand gesture [10][11]. There are many challenges in hand gesture detection such as the model is not able to detect when to start the gesture recognition and stop, secondly, there is a certain set of predefined gestures and the last is the Power consumption of the device running model [10]. These issues were resolved by putting forth a hierarchical framework that would allow convolutional neural network (CNN) designs that functioned offline to perform effectively online utilizing a sliding window technique. The suggested architecture is made up of two models: (8) a detector, which uses a lightweight CNN architecture to find gestures, and (9) a classifier, which uses a deep CNN to identify the motions found. [10]

For the purpose of hand tracking and gesture recognition, additional hardware, such as glove-based input, has been introduced [12]. Dexterity and precision are permitted, but users should work in special rooms like well-lit areas for video cameras or wear a device. Depth sensors like Kinect are likewise a totally famous approach in hand monitoring programs [11]. However, Kinect frameworks will generally concentrate a bigger number of utilizations than on limitation and arrangement strategies. A support vector machine and images of bare hands were used to achieve static gesture recognition [13].

Despite the fact that results vary according to datasets and that extensive pre-processing is required to differentiate background from hand, Rotating and morphologically filtering images are two more pre-processing steps. Convolutional neural networks have also been used to detect hand gestures in real-time [15]. A two-model architecture was proposed in this work.

A five-layer ensemble CNN was used by Yu and Zhang [15] to achieve an accuracy of 0.612. First, the models were pre-trained on the FER-2013 dataset and then finetuned on the Static Facial Expressions in the Wild 2.0 (SFEW). They extracted faces from the labelled movie frames of SFEW by employing a group of three face detectors. Then, they came up with a data perturbation and voting strategy to improve CNN's recognition capabilities. They additionally decided to utilize stochastic pooling layers over max pooling layers referring to its better presentation of their restricted information.

Kahou et al. [16] trained a model on individual video frames and static images with a CNN-RNN architecture. For the video clips, they used the Acted Facial Expressions in the Wild (AFEW) [17] 5.0 dataset, and for the images, they combined the FER-2013 dataset with the Toronto Face Database. They used IRNNs, which are rectified linear units (ReLU), rather than long short-term memory (LSTM) units [18]. The vanishing and exploding gradient problems were dealt with in an easy way by these IRNNs. They were able to achieve a total accuracy of 0.528.

Mollahosseini et al. [19] proposed a network with four Inception layers, max pooling, and two convolutional layers per layer. This network was applied to seven distinct datasets, one of which was the FER-2013 dataset. In addition, they compared the precision of their proposed network to that of an AlexNet [20] network that was trained on the same datasets. They discovered that while the remaining five datasets performed similarly, their architecture performed better on the MMI and FER-2013 datasets. Particularly, the FER-2013 dataset achieved an accuracy of 0.664.

Tools

3.1 Libraries

Making a neural network from scratch for such an application is a really challenging task because of its complexity. The language used for this project in Python. There are many libraries that provide various applications. OpenCV is generally used for image processing whereas TensorFlow and Keras are used for data analysis and machine learning. Numpy is used for mathematical computation. It has been used here because the process requires the conversion of images into two-dimensional matrices.

3.1.1 OpenCV

Computer vision is a way for teaching machines intelligence and train them to view the world as people do. Similar to how humans may analyze a picture of an object and identify it on their own, computer vision entails analyzing images to provide valuable data. In order to implement computer vision, data must be acquired, and a model must then be trained using that data.

The library that is most widely used for computer vision is OpenCV. An open-source computer vision library is called OpenCV. It is based mostly on Python and C/C++. We may use the OpenCV library to create vision-based algorithms like face recognition, pattern recognition, mapping, etc. The OpenCV library can take input in form of images, recorded videos as well as live streams.

3.1.2 TensorFlow

Google created the deep learning library known as TensorFlow. All that tensor flow is a multidimensional array. It processes through an input that is a multi-dimensional array. In computational graphs, nodes in the graphs stand in for operations, while graph edges stand in for tensors. The computation graph is mostly produced via tensor flow. Building and running computational graphs are the two fundamental ideas behind the Tensor Flow software. Python and C++ are only two of the many programming languages supported by TensorFlow. Tensor flow enables quick and precise computer vision processing. Tensor flow is extremely adaptable; it may be used on PCs, on the cloud, and on mobile devices. Both the CPU and the GPU may use it.

3.1.3 Keras

For neural networks, Keras is a high-level deep learning API developed in Python. It facilitates various backend neural network computations and makes neural network implementation simple. Keras is a robust and simple-to-use open-source library that aids in the creation and assessment of deep learning models. The higher-level API of Tensor Flow 2, Keras, is a user-friendly, highly effective interface for addressing problems using contemporary machine learning models that primarily focus on deep learning. It offers crucial abstraction and a foundational building element for creating and delivering machine learning solutions at a fast iteration rate.

Karas features cross-platform capabilities that enable researchers and developers to design deep learning models quickly and effectively.

Keras can be used on TPU and large classes of GPU, and models may be exported for use on browsers and mobile devices. A neural network interface is provided by Keras. It serves as a tensor flow library interface and aids in the creation of tools for working with text and picture data, such as the objective activation function and optimizer, which are frequently used as building blocks in neural networks. By modifying a few lines of code, Keras assists in adjusting the tensor flow data.

3.1.4 Numpy

NumPy is a Python library used for working with arrays. It also has features for running in the domain of linear algebra, Fourier remodel, and matrices. NumPy become created in 2005 by using Travis Oliphant. It is an open-source undertaking, and you can use it freely. NumPy stands for Numerical Python. In Python we've lists that serve the motive of arrays, however, they're sluggish to the system. NumPy objectives are to provide an array object that is as much as 50x faster than traditional Python lists. Arrays are very frequently utilized in information science, in which speed and sources are very important.

3.2 CNN

CNN is the most widely used neural network for language and image processing (Convolutional Neural Network). It is a crucial component of deep learning and artificial intelligence. An input layer, hidden layers, and an output layer are all present in a standard neural network. While the hidden layers process these inputs, the input layer takes inputs in a variety of formats. The results of the computations and extractions are subsequently delivered by the output layer. Each of these layers has neurons that are linked to neurons in the layer below, and every neuron in each layer is weighted differently. This indicates that we are not assuming anything regarding the data being supplied into the network. Convolutional neural networks, on the other hand, operate differently because they regard data as spatial. Neurons aren't coupled to every neuron in the preceding layer; rather, they are only connected to those that are nearby and have all the same weight. The network maintains the spatial component of the data set due to the links' simplicity. such a difficult procedure. For improved processing and comprehension. The convolutional layer and the pooling layer are two of the layers that set it apart. But it will also feature a ReLU or rectified linear unit layer and a fully connected layer, much like previous neural networks. In order to ensure non-linearity as the data passes through each layer of the network, the ReLU layer serves as an activation function. Without it, the dimensionality that we wish to keep in the data being fed into each layer would be lost. In the meanwhile, the completely linked layer enables us to classify our data set.

Methodology

4.1 Dataset

The well-known FER 2013 dataset was used in the Kaggle competition. The data must be prepared for input to the CNN because there are some issues with this dataset as discussed below. So, because the model's input should be an array of numbers, the images must be converted into arrays. The images used for training should be free of the issues listed above. Thus, the 35,000 images in the FER 2013 dataset were manually filtered, and 7,074 images from five classes were picked: 966 for angry, 859 for fear, 2477 for happy, 1466 for neutral, and 1326 for sad. Apart from all these, the other dataset for hand gesture recognition was also taken from Kaggle. The dataset consisted of many hand gestures but for training, this model, data for also four gestures were used. This data included, 1000 each for fist, thumbs up, stop, and peace.

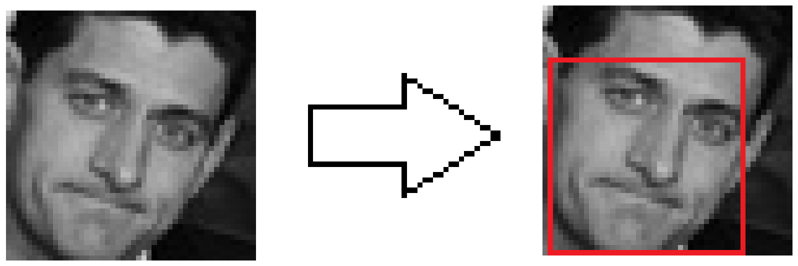


Fig. 4 Main Features of the face detected

For training the machine learning model, only a useful part of the data must be used. Firstly, just the main features of the face should be captured and stored. After, detecting that feature, the photo is cropped and then converted into an array of pixels. Each array consisted of 48 by 48 pixels. This is done by the array module in the NumPy library. Fig. 4 shown above displays the main features captured in each photo of the dataset. In the case of hand gestures, the photos of the dataset are converted into grayscale directly (fig. 5).

Fig 5: Gray-scaled Image

4.2 Convolution Neural Network

CNN Model for the facial expressions is divided into four stages. The size of the input image is reduced at the end of each phase. The first three phases all use the same layers, which begin with a convolution and end with a dropout. The first phase of the model has an input layer for an image of size 48 × 48 (height and width in pixels) and convolution is performed on this input. The number of convolution layers in Table 3.1 is the same for all convolution layers except the number of kernels. In the first phase, there are 64 kernels. The next layer's inputs are then obtained through batch normalization. Convolution and batch normalization are repeated in the following layers. In the following layer, max pooling is performed with a pool size of 2 x 2, resulting in an output size of 24 x 24. Dropout occurs next at a rate of 0.35. The second phase contains 128 kernels and has a dropout rate of 0.4. In the second phase, max pooling produces an output of size 12 x 12. The third phase contains 256 kernels with a dropout rate of 0.5. In the third phase, maximum pooling is used. Layers of input and conversion batch normalization Pooling Dropout Dense 17 reduces the output size to 6 6. The final phase begins with a flattened layer and continues with dense and output layers. The data must be a one-dimensional array to be classified into the five emotions. The flattened layer reduces two-dimensional data to a single-dimensional array. The SoftMax function is applied to the flattened output by the dense layer. Then, batch normalization is done, and the output layer gives the class probabilities.

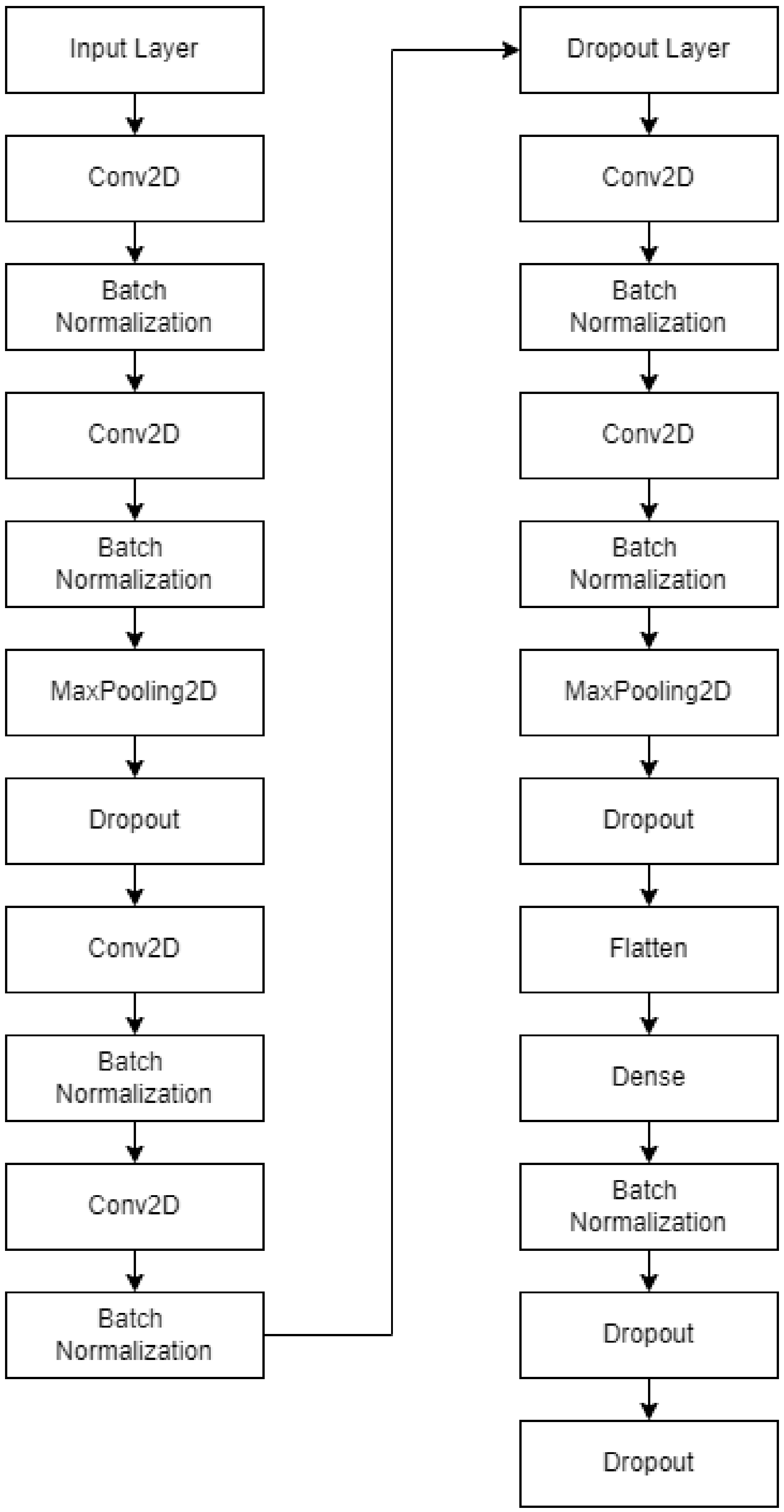


Fig 6: CNN of facial recognition

The hand gesture network is described as followed. The input is given as a 64 by 64 matrix of pixels. The first layer is a convolutional layer with a filter size of (5,5) consisting of 32 filters with the default stride and padding of size 1. It is followed by a Max pooling layer with a pool size of (2,2). Pooling has been added as it reduces the height and width of the samples and thereby reduces the computation time. The second convolutional layer has the size of 64 filters, with a (3,3) kernel size. the third and 4th convolutional layers consist of 96 filters of kernel size (3,3). These convolutional layers are immediately followed by Max pooling layers of pool size (2,2) and a stride of 2. Flattening of the output is done to convert the feature map in do a single column before passing it to the final dense layer This flattened output is passed through a fully connected layer of 512 nodes. The ReLU activation function is used for all the layers, which essentially gets applied for the entire model, and the SoftMax activation function is used after the final dense layer.

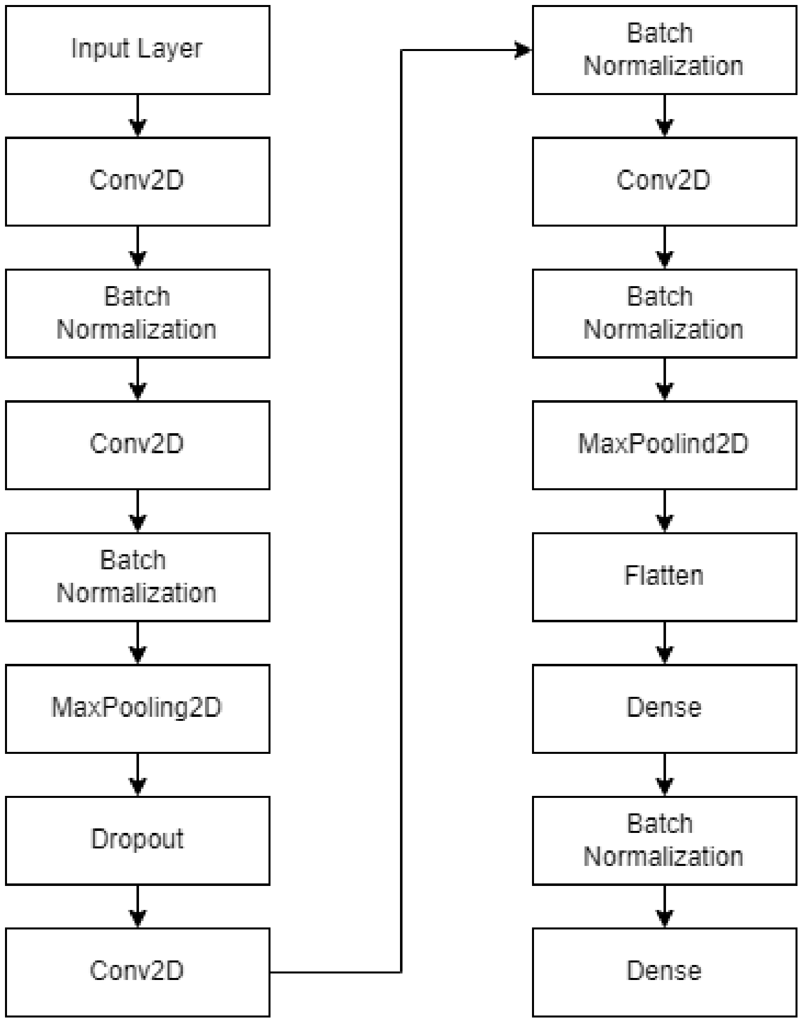
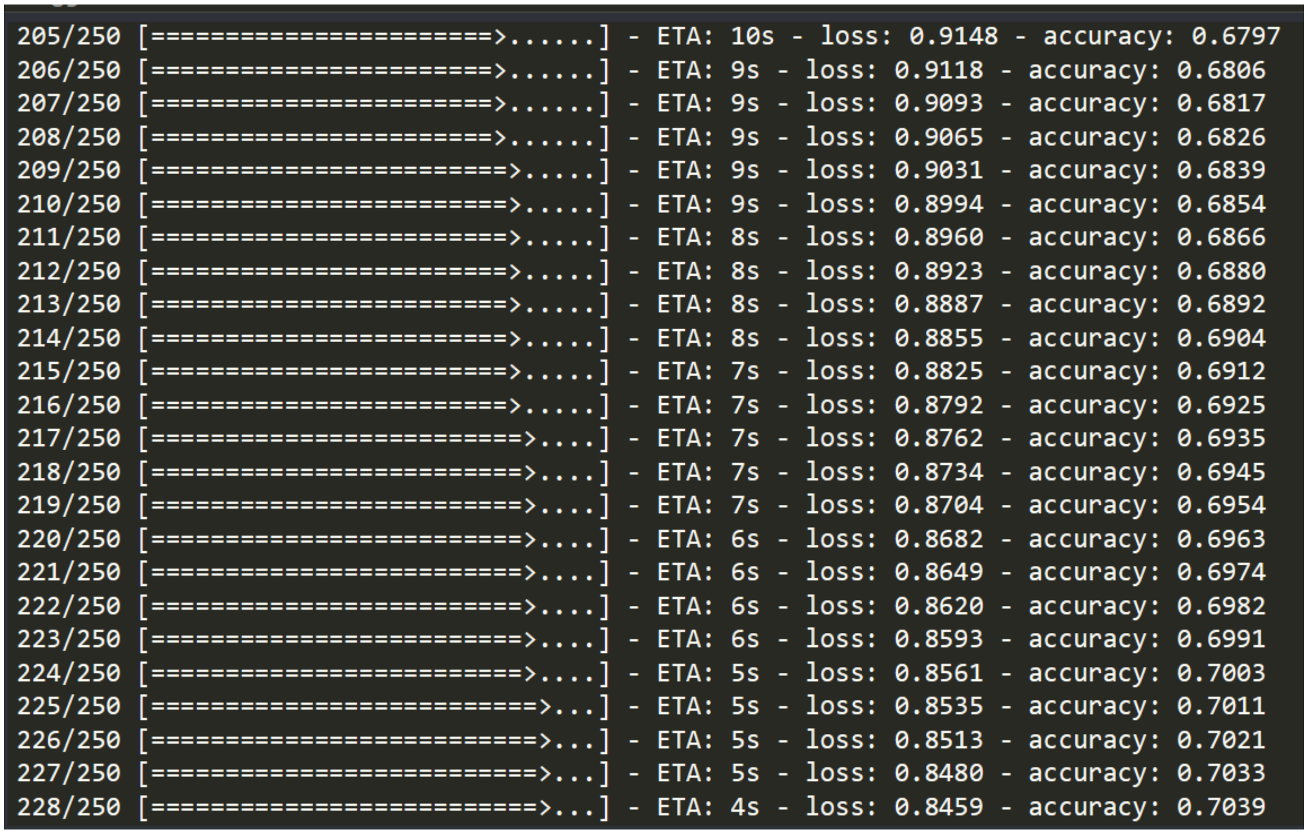


Fig 7: CNN for Hand Gestures

4.3 Training the Model

The compilation of both models was done using Adam optimizer and the metrics used were accuracy. The dataset taken from Kaggle was already divided into training and testing data. The ratio of the division was approximately 3:1. 75% of the data was used to train the model and the rest 25% was used for testing. The Learning Rate (LR) is a training parameter that determines how quickly the model weights are calculated. A high LR can cause the model to converge too quickly while a small LR may lead to more accurate weights (up to convergence) but takes more computation time. The number of epochs is the number of times a dataset is passed through the NN forward and backward.



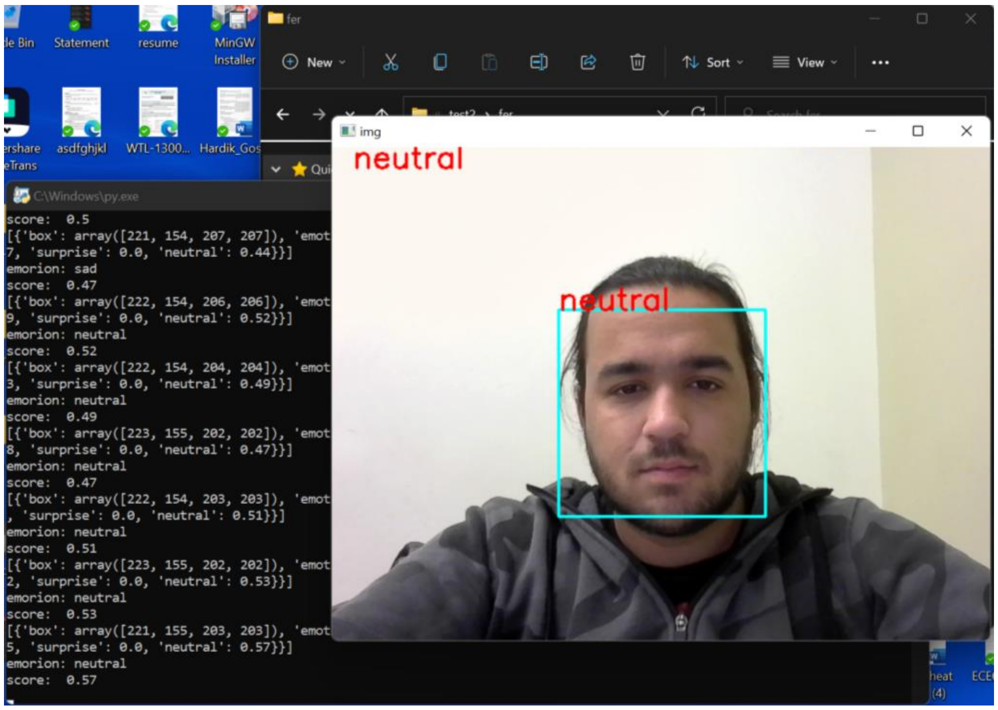
```
205/250 [======================>......] - ETA: 10s - loss: 0.9148 - accuracy: 0.6797
206/250 [======================>......] - ETA: 9s - loss: 0.9118 - accuracy: 0.6806
207/250 [======================>......] - ETA: 9s - loss: 0.9093 - accuracy: 0.6817
208/250 [======================>......] - ETA: 9s - loss: 0.9065 - accuracy: 0.6826
209/250 [=======================>.....] - ETA: 9s - loss: 0.9031 - accuracy: 0.6839
210/250 [=======================>.....] - ETA: 9s - loss: 0.8994 - accuracy: 0.6854
211/250 [=======================>.....] - ETA: 8s - loss: 0.8960 - accuracy: 0.6866
212/250 [=======================>.....] - ETA: 8s - loss: 0.8923 - accuracy: 0.6880
213/250 [=======================>.....] - ETA: 8s - loss: 0.8887 - accuracy: 0.6892
214/250 [=======================>.....] - ETA: 8s - loss: 0.8855 - accuracy: 0.6904
215/250 [=======================>.....] - ETA: 7s - loss: 0.8825 - accuracy: 0.6912
216/250 [=======================>.....] - ETA: 7s - loss: 0.8792 - accuracy: 0.6925
217/250 [========================>....] - ETA: 7s - loss: 0.8762 - accuracy: 0.6935
218/250 [========================>....] - ETA: 7s - loss: 0.8734 - accuracy: 0.6945
219/250 [========================>....] - ETA: 7s - loss: 0.8704 - accuracy: 0.6954
220/250 [========================>....] - ETA: 6s - loss: 0.8682 - accuracy: 0.6963
221/250 [========================>....] - ETA: 6s - loss: 0.8649 - accuracy: 0.6974
222/250 [========================>....] - ETA: 6s - loss: 0.8620 - accuracy: 0.6982
223/250 [========================>....] - ETA: 6s - loss: 0.8593 - accuracy: 0.6991
224/250 [========================>....] - ETA: 5s - loss: 0.8561 - accuracy: 0.7003
225/250 [=========================>...] - ETA: 5s - loss: 0.8535 - accuracy: 0.7011
226/250 [=========================>...] - ETA: 5s - loss: 0.8513 - accuracy: 0.7021
227/250 [=========================>...] - ETA: 5s - loss: 0.8480 - accuracy: 0.7033
228/250 [=========================>...] - ETA: 4s - loss: 0.8459 - accuracy: 0.7039
```

Fig 8: Training of Hand gesture CNN model

4.4 Real-Time Facial expressions and Hand gestures Analysis

After training the model, it was stored in an H5 file. An H5 is a Hierarchical Data Format (HDF) that is used to store large amounts of data. Its multidimensional arrays are used to store large amounts of data. The format is primarily used to store well-organized scientific data for easy retrieval and analysis. After the model was stored, in a separate code, it was loaded back so as to save the time of training the model again and again.

In the face expressions recognition, using OpenCV live video was captured. The video was analyzed per frame. In each frame, firstly the image was converted into grayscale and then the face was detected using the haarcascade frontalface classifier provided along with the OpenCV library. After detecting the face, just the face was cropped as only the required area must be given as an input for prediction. After cropping the face, it was converted into 48 by 48 pixels using the resize () function. After getting the predicted results, it was overlayed onto the original frame that was captured and displayed as output to the user.



```
score:  0.5
[{'box': array([221, 154, 207, 207]), 'emot
7, 'surprise': 0.0, 'neutral': 0.44}}]
emorion: sad
score:  0.47
[{'box': array([222, 154, 206, 206]), 'emot
9, 'surprise': 0.0, 'neutral': 0.52}}]
emorion: neutral
score:  0.52
[{'box': array([222, 154, 204, 204]), 'emot
3, 'surprise': 0.0, 'neutral': 0.49}}]
emorion: neutral
score:  0.49
[{'box': array([223, 155, 202, 202]), 'emot
8, 'surprise': 0.0, 'neutral': 0.47}}]
emorion: neutral
score:  0.47
[{'box': array([222, 154, 203, 203]), 'emot
, 'surprise': 0.0, 'neutral': 0.51}}]
emorion: neutral
score:  0.51
[{'box': array([223, 155, 202, 202]), 'emot
2, 'surprise': 0.0, 'neutral': 0.53}}]
emorion: neutral
score:  0.53
[{'box': array([221, 155, 203, 203]), 'emot
5, 'surprise': 0.0, 'neutral': 0.57}}]
emorion: neutral
score:  0.57
```

Fig 9: *Screenshot of the live video stream*

Similarly, in hand gesture recognition, using OpenCV live video was captured. The video was analyzed per frame. In each frame, firstly the image was converted into grayscale and then it was checked whether there is any actual hand present or not. If it detects the presence of a hand, it converts the input frame into 48 by 48 pixels using the resize () function. After getting the predicted results, it was overlayed onto the original frame that was captured and displayed as output to the user.

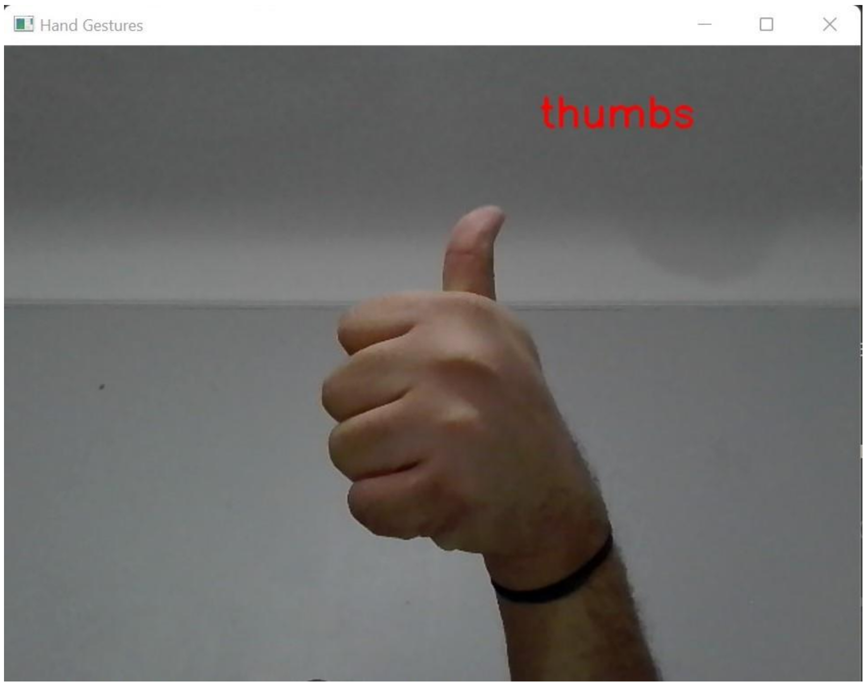


Fig 10: Screenshot of the live video stream

Results

The parameters used for both models were different. The parameters for the models that were defined for the model were inspired by existing research in this field. The optimizer and the metrics chosen for these models were ‘Adam’ and ‘Accuracy’. Parameters like the learning rate and the number of epochs were set using the hit-and-trial method. The parameters with the best convergence were chosen.

The parameters used for both models are given below in Tables 1 and 2.

Parameters	Values
Learning Rate	0.001
Batch Size	64
Number of Epochs	100

Table 1: Parameters for Facial Expressions Recognition model

Parameters	Values
Learning Rate	0.001
Batch Size	32
Number of Epochs	4000

Table 2: Parameters for Hand Gesture Recognition model

After training the model with these parameters, the maximum accuracy got from the facial expression model was approximately 72% and the accuracy of the hand gesture model was 80%.

Comparing both plots “training loss vs. training epoch” and “accuracy vs. training epoch” for both the models. There are three possible outcomes for the learning curves:

• Underfitting: An underfit model is one that is unable to learn the training data well. The training loss's learning curve can be utilized to spot an underfit model. The curve could be a flat line or a series of noisy values with significant loss, indicating that the model did not learn the training dataset. An underfit model may also be indicated by a training loss that is decreasing and continues to decrease until the end of the plot.

• Overfitting: A model is said to be overfitting if it has learned the training dataset too well, including statistical noise or random fluctuations in the training dataset. Overfitting is challenging because the more specialized a model is for training data, the less successful it can generalize to new data, resulting in a higher generalization error. The plot of learning curves reveals overfitting if the plot of training loss continues to decrease with the epoch. The validation loss plot then drops to a point before rising again.

• Optimal fitting: The purpose of the learning algorithm is to find a decent match between an overfit and an underfit model. The plot of learning curves displays a good match if the plot of training loss decreases to a point of stability and the plot of validation loss decreases to a point of stability with a small gap with the training loss.

By looking at the plots of Loss vs Epoch for both models, we can say that there is very less generalization gap and hence signifying that our model is better for the analysis of the given dataset

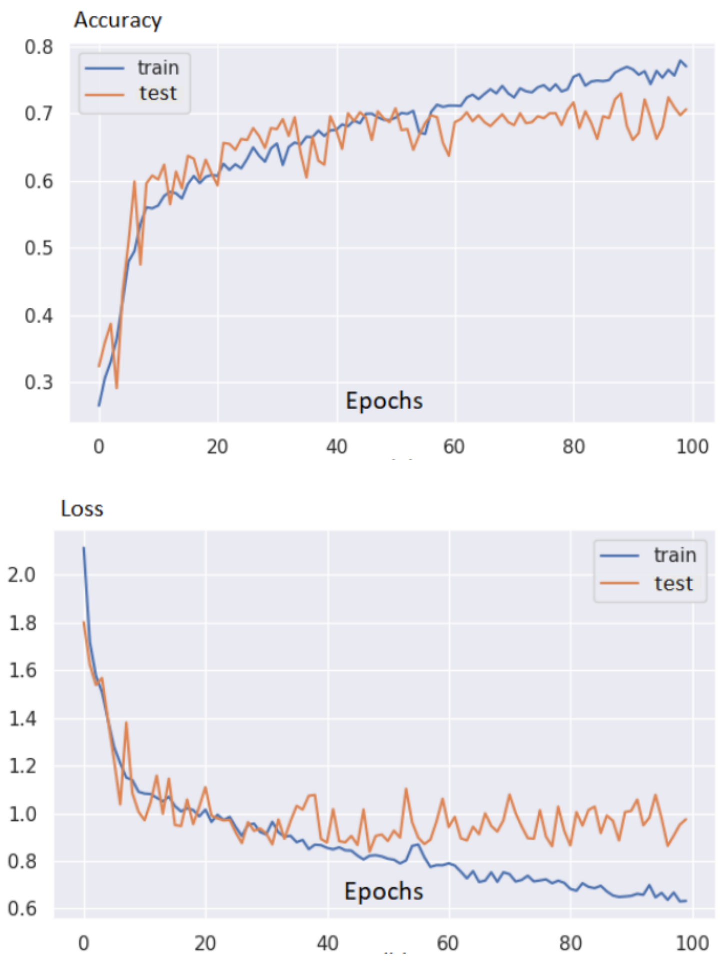


Fig 11: Facial expression CNN model

These are some of the results from the live execution of the code.

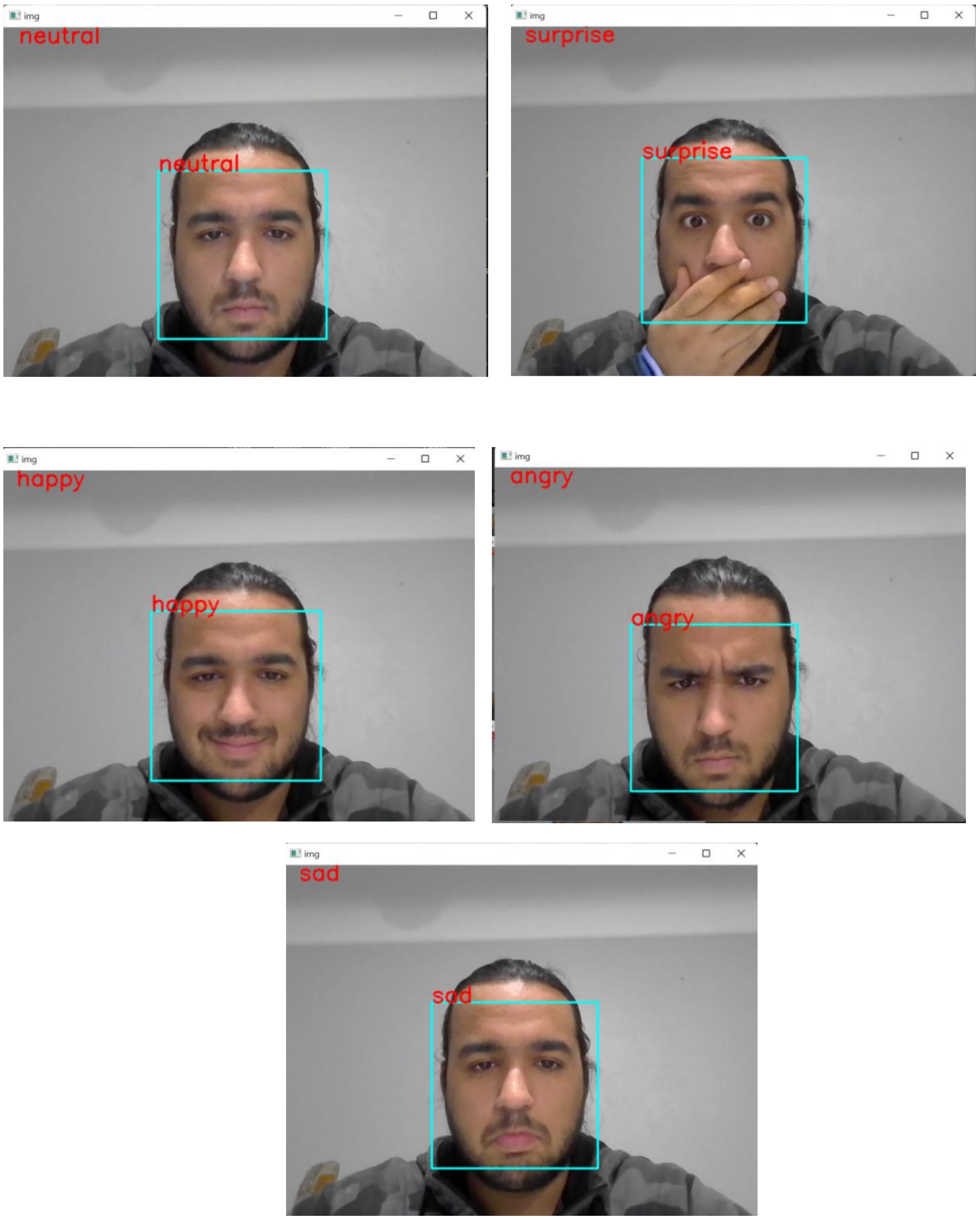


Fig 13: Prediction results of the facial expression recognition model

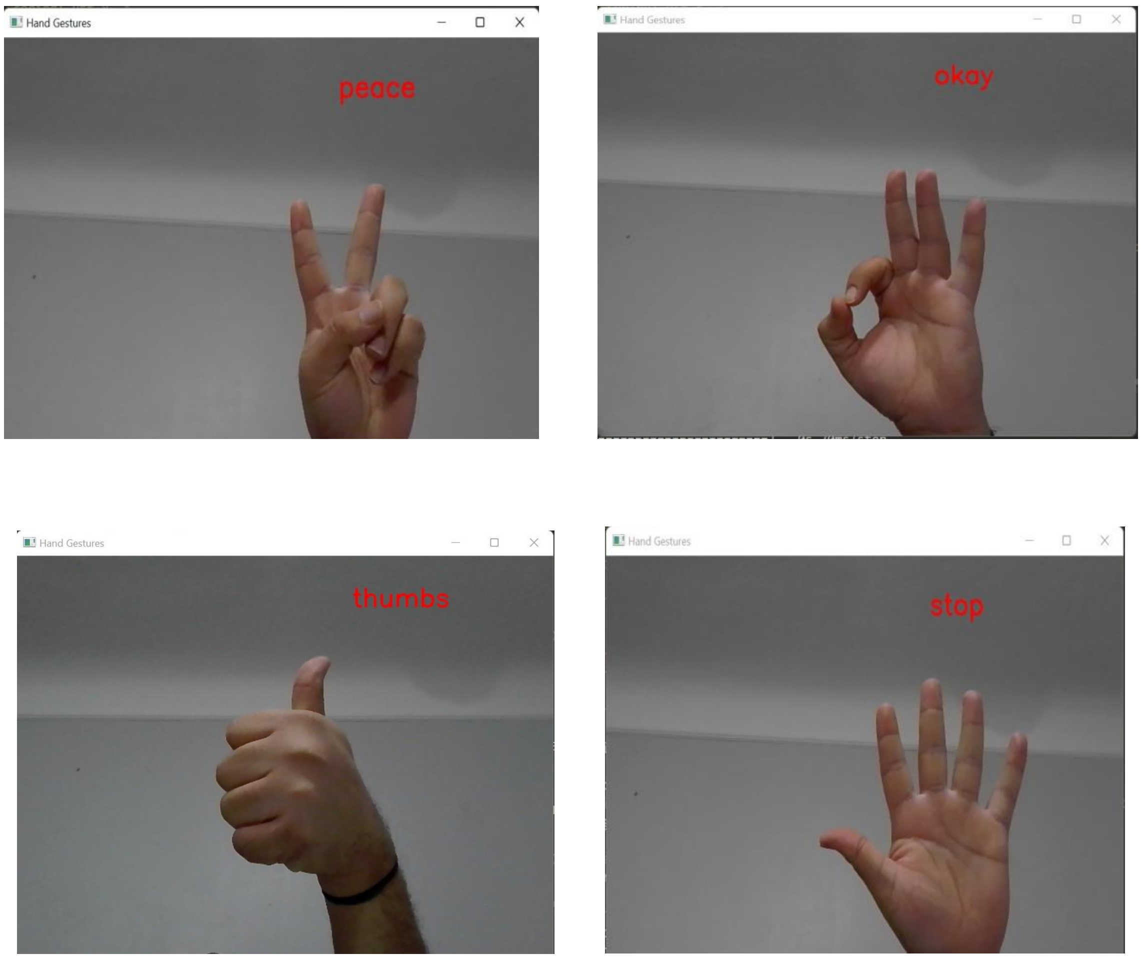


Fig 14: Prediction results of the hand gesture recognition model

Conclusion

In this project, I successfully used convolutional neural networks to build two separate models for facial expressions recognition and hand gesture recognition. By tuning the parameters with the help of previous work, along with the hit and trial method, the maximum accuracy received was 72% for the facial expression model and 80% for the hand recognition model. Five of the facial expressions including angry, happy, sad, neutral, and surprised were recognized by the camera robustly. On the other hand, four of the hand gestures were recognized including, stop, thumbs up, peace, and okay. When testing these trained models on live video feeds, it caused more inaccuracy than expected while detecting the expressions and gestures. This problem can be caused due to processing power or frames drops. This issue can be addressed and resolved in future work.

References

[1] Fischer, K., Jung, M., Jensen, L. C., & aus der Wieschen, M. V. (2019, March). Emotion expression in HRI–when and why. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (pp. 29-38). IEEE.

[2] Cross, E. S., & Ramsey, R. (2021). Mind meets machine: towards a cognitive science of human–machine interactions. *Trends in Cognitive Sciences*, *25*(3), 200-212.

[3] Phutela, D. (2015). The importance of non-verbal communication. *IUP Journal of Soft Skills*, *9*(4), 43.

[4] Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M., & Anderla, A. (2019, March). Credit card fraud detection-machine learning methods. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)* (pp. 1-5). IEEE.

[5] Ilyas, B. R., Mohammed, B., Khaled, M., & Miloud, K. (2019, November). Enhanced face recognition system based on deep CNN. In *2019 6th International Conference on Image and Signal Processing and their Applications (ISPA)* (pp. 1-6). IEEE.

[6] Gadekallu, T. R., Alazab, M., Kaluri, R., Maddikunta, P. K. R., Bhattacharya, S., & Lakshmanna, K. (2021). Hand gesture classification using a novel CNN-crow search algorithm. *Complex & Intelligent Systems*, *7*(4), 1855-1868.

[7] Rawal, N., & Stock-Homburg, R. M. (2022). Facial emotion expressions in human–robot interaction: a survey. *International Journal of Social Robotics*, *14*(7), 1583-1604.

[8] Jiang, D., Li, G., Sun, Y., Kong, J., & Tao, B. (2019). Gesture recognition based on skeletonization algorithm and CNN with ASL database. *Multimedia Tools and Applications*, *78*(21), 29953-29970.

[9] Mehendale, N. (2020). Facial emotion recognition using convolutional neural networks (FERC). *SN Applied Sciences*, *2*(3), 1-8.

[10]Khan, S., Javed, M. H., Ahmed, E., Shah, S. A., & Ali, S. U. (2019, March). Facial recognition using convolutional neural networks and implementation on smart glasses. In *2019 International Conference on Information Science and Communication Technology (ICISCT)* (pp. 1-6). IEEE.

[11]Köpüklü, O., Gunduz, A., Kose, N., & Rigoll, G. (2019, May). Real-time hand gesture detection and classification using convolutional neural networks. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)* (pp. 1-8). IEEE.

[12]J. Suarez and R. R. Murphy, "Hand gesture recognition with depth images: A review," 2012 IEEE RO-MAN: The 21st IEEE International

[13]D. J. Sturman and D. Zeltzer, "A survey of glove-based input," in IEEE Computer Graphics and Applications, vol. 14, no. 1, pp. 30-39, Jan. 1994, doi: 10.1109/38.250916.

[14]D. K. Ghosh and S. Ari, "Static Hand Gesture Recognition Using Mixture of Features and SVM Classifier," 2015 Fifth International Conference on Communication Systems and Network Technologies, Gwalior, 2015, pp. 1094-1099, doi: 10.1109/CSNT.2015.18.

[15]Kopuklu, Okan & Gunduz, Ahmet & Köse, Neslihan & Rigoll, Gerhard. (2019). Real-time Hand Gesture Detection and Classification Using Convolutional Neural Networks.

[16]Zhiding Yu & Cha Zhang. (2015). Image based Static Facial Expression Recognition with Multiple Deep Network Learning. Proceedings of the 2015 ACM on International Conference on Multimodal Interaction. 435- 442.

[17]Ebrahimi Kahou, S., Michalski, V., Konda, K., Memisevic, R., and Pal, C. (2015). Recurrent neural networks for emotion recognition in video. Proceedings of the 2015 ACM on International Conference on Multimodal Interaction. 467-474. ACM.

[18]Abhinav Dhall, Roland Goecke, Simon Lucey, Tom Gedeon. (2012). Collecting Large, Richly Annotated Facial-Expression Databases from Movies, IEEE Multimedia, 19(3):3441, July 2012

[19] Q. V. Le, N. Jaitly, and G. E. Hinton. (2015). A simple way to initialize recurrent networks of rectified linear units. arXiv preprint arXiv:1504.00941.