Islamic University of Technology (IUT)

Department of Computer Science and Engineering (CSE)

Vision-based Therapeutic System Involving Finger Exercise

Authors

Nafis Saami Azad - 170042007

Al Muhaimin - 170042010

Maliha Mehzabin Zoyee - 170042048

Supervisor

Mohammad Ridwan Kabir

Lecturer, Department of CSE

A thesis submitted to the Department of CSE

in partial fulfillment of the requirements for the degree of B.Sc.

Engineering in SWE

Academic Year: 2017-18

April - 2022

Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by Nafis Saami Azad, Al Muhaimin and Maliha Mehzabin Zoyee under the supervision of Mohammad Ridwan Kabir, Lecturer of the Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh. It is also declared that neither of this thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Authors:

Nafis

Nafis Saami Azad
Student ID - 170042007

Al Muhaimin
Student ID - 170042010

Zoyee

Maliha Mehzabin Zoyee
Student ID - 170042048

Supervisor:

Mohammad Ridwan Kabir

Lecturer

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

Acknowledgement

We would like to express our grateful appreciation for **Mohammad Ridwan Kabir**, Lecturer, Department of Computer Science & Engineering, IUT for being our mentor. His drive, thoughts, and insights have been vital to this research. This research would not have been accomplished without his help and supervision. His important opinion, time, and input were offered throughout the thesis work, from the first phase of thesis topic introduction, subject selection, algorithm proposal, revision, and project implementation and finalization, which assisted us in doing our thesis work properly. We owe him a great debt of gratitude.

We are also grateful to **Md. Ishrak Abedin**, Lecturer, Department of Computer Science & Engineering, IUT for his valuable inspection and suggestions on our work and data collection process, and also helping us in building the PICS interface. It would have been extremely difficult to collect and annotate data according to our requirements without his assistance.

We would also like to thank individuals who gave their time to help us to collect data so that we could create our own dataset.

Abstract

This work aims to create a system that will allow patients to receive physiotherapy at home. We focus on two key parts, a vision based approach, and rehabilitation therapy primarily for stroke patients with upper limb disability through finger exercises. Our hardware consists of only using a HD webcam or a camera. For our hand detection, we are using three deep learning models. The first model is trained using Google's Mediapipe hands to find the hand landmark from an image, and using the landmarks to recognize the therapeutic hand gestures. To train our model, we also collected an extensive dataset for the pinching finger exercises with different environmental conditions, distances from camera and orientations. The data we collected have been used to create 3 deep learning models capable of detecting therapeutic hand gestures from real-time video feeds. In this report, we have discussed the methodologies we used for data collection, the types of models we built and their architecture along with their strong points and shortcomings. Lastly, we elaborated on the future works that we intend to perform.

Contents

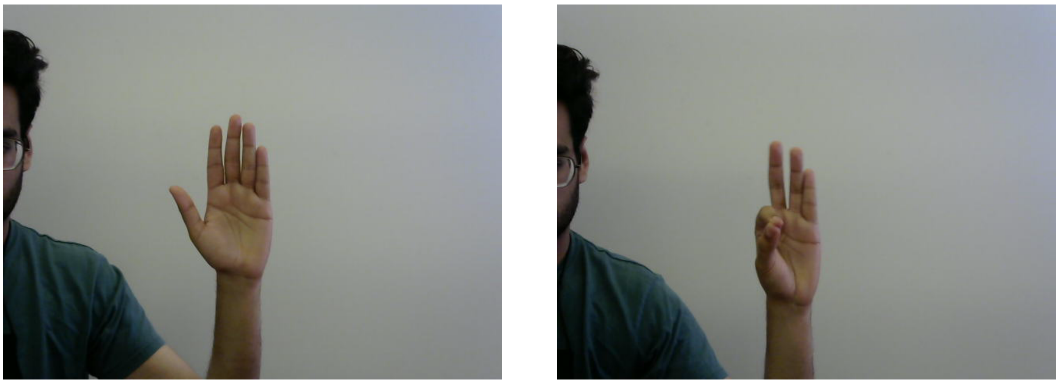
List of Figures

List of Tables

1 Introduction

1.1 Overview

Rehabilitation is the process of nursing a person back to a healthy and normal life through special therapeutic exercises and training. Strokes can occur due to injuries in different parts of the brain or spinal cord [1]. Strokes cause either partial or full paralysis, depending on the region where the stroke occurred. For partial paralysis patients, it is possible for them to regain some control of their upper limbs through therapeutic hand exercises. For our work, we primarily aim to focus on the Pinching finger exercise. We plan to incorporate other finger exercises in our future work. The pinching exercise is performed by moving your index finger from fully upright position to touching the thumb, while keeping all other fingers fully upright. Our work aims to automate this exercise in a cost efficient environment so that patients who cannot go to clinics or health centers can effectively perform them at home.



(a) Open-hand posture

(b) Pinching Posture

Figure 1.1: Smooth transitioning dataset requirement

Since we want to be able to detect only a handful of very specific gestures (like pinching) very accurately, we needed to create our own dataset. The existing datasets that we have found did not meet our very specific requirements. We require datasets that have images that transition smoothly from a open-hand posture (figure 1.1a) to the pinching posture (figure 1.1b). So we opted to build an extensive dataset by collecting over 20,000 pictures for the pinching exercise and use feature engineering to generate features. We then used our dataset to train 3 deep learning models that can accurately detect the pinching exercise.

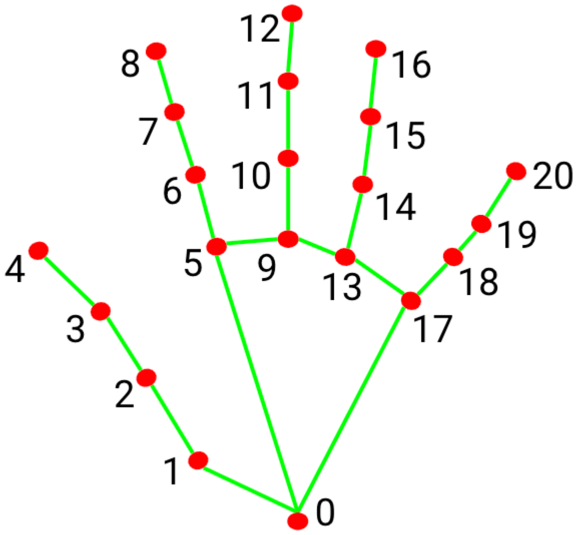


Figure 1.2: Hand landmarks

Our first approach uses Google's *Mediapipe Hands* for hand segmentation and hand landmarks detection. Hand landmarks are specific virtual points that are generated on the hand in an image, as is in figure 1.2. Each of these points have positional (x, y) coordinates that can be used to accomplish certain tasks. Our first model is a very basic convolutional neural network (CNN) model with only 2 dense layers. Our approach consists of firstly detecting the hand landmarks from the image dataset, storing the positional values of each landmark coordinate into a `.csv` file along with their label, and finally using that `.csv` file to train a CNN model capable of detecting the hand gestures.

For our second approach, we used an image classifier that can classify the images of the different hand postures. An image classifier is a deep-learning model that is capable of differentiating between two static images, and can understand which class each of the images belong to. A class of an image is the category or type of the image. For example, cats and dogs represent two different classes of images.

For this approach, the hand was segmented from each image and it was fed into the image classifier model for training. The model architecture consists of 3 pairs of *Conv2D* and *MaxPooling* layers, followed by a *Flatten* layer and finishing it with two more *Dense* layers, with the output layer having the *softmax* activation function.

The third approach is an ensemble model made from the two previous models. Ensemble learning is a general meta approach to machine learning that seeks better predictive performance by combining the predictions from multiple models [2]. Both of our previous models used the *softmax* activation function at the last layer. We took the per-class prediction value for each model, and created an average value for each class. This per-class average value is considered as the output of the ensemble model. We evaluated our test sets based on the average values for each-class and the predicted output class was the class with the highest value.

1.2 Problem Statement

After the recent event of the pandemic, the whole universe has changed along with its way of living life. The majority of actions and works are transferred into online based systems. Keeping that in mind, it became increasingly difficult for patients to get to the hospital for their check up and other necessities. Most importantly, the paralysed patients who need physiotherapy on a regular basis faced a lot of inconvenience during this time. Even if we skip the pandemic part, it is always difficult and inconvenient for paralyzed patients to get out of their homes and travel all the way to the hospital to get therapy and check ups. To solve these issues, we wanted to come up with a workaround to help these people and make their life easier.

1.3 Motivation & Scopes

1.3.1 Motivation

Patient rehabilitation systems using computer vision is a relatively unexplored topic. Even after the advances in the field of Computer Vision, very little significant work has been done in this topic. There is also a lack of viable large datasets regarding hand rehabilitation therapy. Most of the papers we have reviewed have either used a depth camera [3] or some sort of data glove or device [4]. Depth cameras and data gloves are not cost efficient solutions. Depth cameras are expensive and are not commonly used in everyday situations. Data gloves need to be specifically manufactured, and it is difficult to mass produce them, and they can also be expensive to make. Since our goal is to allow patients to perform rehabilitation exercises at home in a cost-effective way, we will only be working with standard webcams. Webcams are widely available, used in daily affairs and are not very expensive. This is what motivated us to use webcams instead of any other alternative solutions.

1.3.2 Scope

The main objective of our work is ensuring proper rehabilitation of stroke patients with upper limb disability by allowing them to perform therapeutic finger exercises accurately at home. To meet the objective, we need to work with computer vision and image processing, since we will be using a standard camera to capture video-feed and also work with deep-learning models so that our system can measure the accuracy of the therapeutic finger exercise being performed.

The scope of our work contains capturing and processing of input from a camera and working with different types of deep-learning models to accurately understand the pinching exercise from a real-time video-feed.

1.4 Research Challenges

The biggest challenge that we had to face during our research was to create the dataset in the midst of the COVID-19 situation. Since data collection is a physical task, we had to face a lot of issues due to the restrictions and lock-downs. Finding people willing to donate time for us was also an issue.

Another challenge we faced was data annotation. Since we were looking to collect over 20,000 images, we needed an efficient way to annotate them. Manual annotation would have been very redundant and time-consuming. So we opted to create an interface that would automatically annotate the image only we took it. Creating this interface was a big challenge for us.

For training the model, we had to select a lightweight model, as we are using real time video feed. We had to face some problems finding the perfect model for our purpose. Then again, the accuracy of the models were also a big concern for us. Selection of vital features for posture detection was another crucial factor here. But, the biggest challenge was to train the model with such a huge number of dataset given that we had limited resources.

1.5 Thesis Outline

In Chapter 1 we have discussed about our study in a simple and straightforward manner. We have given an overview of what we are doing and how we aim to do it. Chapter 2 deals with the necessary literature review for our study and there development so far. In Chapter 3 we have stated the skeleton of our proposed method, the different types of models we used and their architecture. Chapter 4 shows the results and comparative analysis of of the different models, along with their pros and cons. We also gave a rationale on why we think the models performed the way they performed. Chapter 5 contains the summary of our work and the future works we tend to perform. The final segment of this study contains all the references and credits used.

2 Literature Review

2.1 Rehabilitation

Stroke is the largest cause of adult disability globally, with an estimated 16 million new cases each year.[5] As more time passes, the rate of stroke patients is increasing at an alarming rate.

Rehabilitation is a very unique but also a common field to work on. Specifically for the patients who have suffered from stroke and suffer from upper limb disability. It has been indicated that certain exercises for hands helps in increasing mobility and usage of hand and finger. Various smoothness criteria have been utilized in research with stroke victims in the past. It is clear that the measures chosen are not always properly justified, and sometimes erroneous metrics were introduced. It was discovered that 31 distinct metrics were used that quantify the smoothness of stroke patients' reaching movements [6]. Because the measurements will be utilized with stroke patients, it is important to examine if they can capture changes in smoothness over time during stroke patients' rehabilitation. Another significant finding is the distinction in velocity profiles of reach-to-grasp and pointing motions.[6] The knowledge that we gained from here is that, the metrics that are described here helps us to figure out if a patient is recovering or not. The result analysis of our future work lies along with defining such kind of metrics.

Recently many robotics arm/hand based coordination system has been invented in the field of study as it helps rigorously in the rehabilitation process of upper limb disability based stroke patients. "Multi-finger coordination in healthy subjects and stroke patients: a mathematical modeling approach" [7] is the next journal paper that we reviewed. In this paper the authors made a tool using which they can completely characterize the objective of spatial and temporal aspects of hand movement in stroke patients. The authors conducted hand opening and closing movement experiments in 12 healthy volunteers and 14 stroke survivors. The extension of metacarpophalangeal and proximal interphalangeal joints of all fingers were taken. Then a four-parameter hyperbolic tangent function was computed and mathematically characterized. And for analysis purposes they used the coefficient of determination and root mean square error. Intraclass correlation coefficient (ICC) and test-retest errors were used to measure test-retest reliability. Analysis of probable changes in parameters describing angular and temporal features of hand kinematics and inter-joint, inter-digit coordination was used to compare the performances of healthy controls and stroke patients. The value of ICC was abnormally higher for the stroke patients than that of normal people. This result shows us the difference between the finger movements of normal people and stroke survivors, which was a useful intel for the implementation of our experiment.

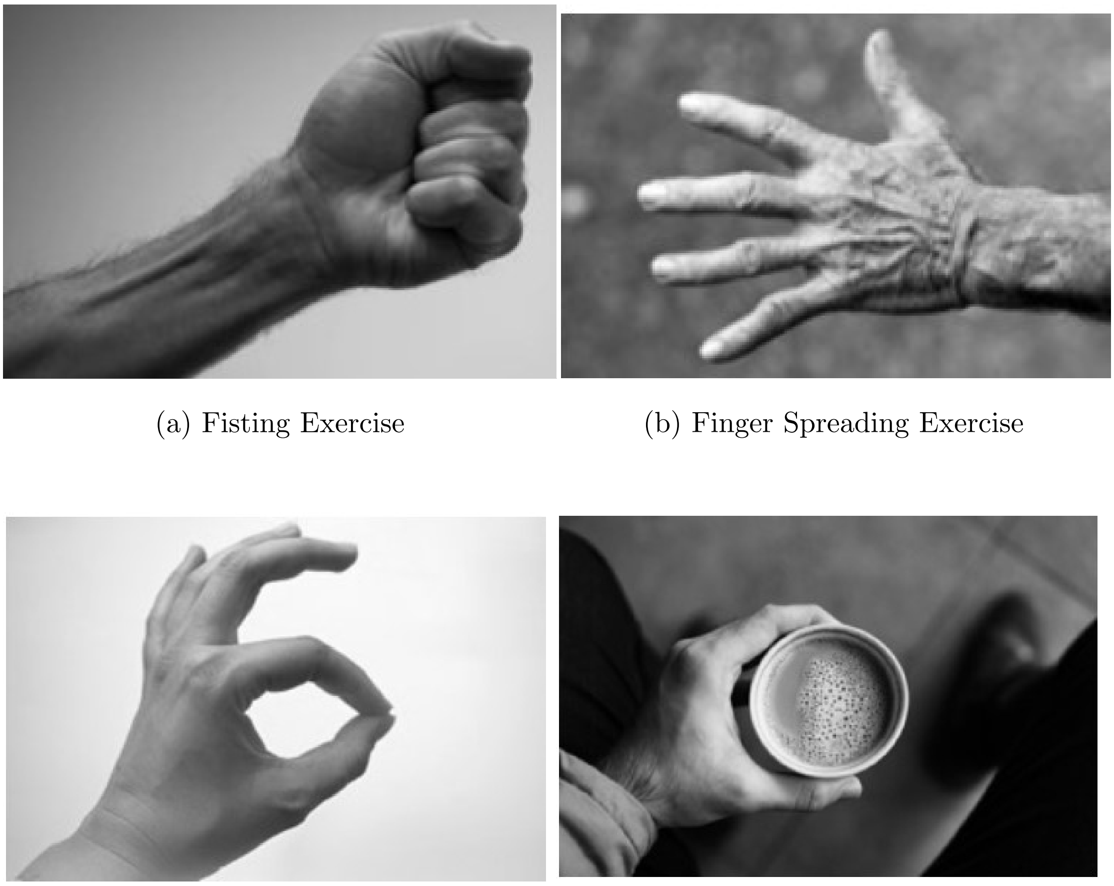
Gripping and pinching activities need the use of one's fingers. Following an injury or even a stroke, a meticulous rehabilitation program with the occasional presence of a specialist is required. A major part of rehabilitation is providing patients with a home exercise routine. Typically, no technological aids are used in a home-based workout regimen.

Certain device have been built and referenced in case of rehabilitation for stroke patient with upper limb disability. The next paper that we found implements this functionality. The name of the paper is "Development of a Low-cost Glove for Thumb Rehabilitation: Design and Evaluation"[8]. The authors in this paper invented a glove type wearable device which helps with the movement of the thumb and detecting and calculating the actions using an android device. This device can be worn by the patient. While wearing the device they conducted exercises and saw their own progress using their smartphone. It enables home-therapy and alongside that it decreases the cost too. This device is also a very helpful tool for the therapists to detect the progress of the patients quickly and efficiently. The benefit of this tool is that it is portable and easy to use and cost efficient. But our implementation uses real time video and the whole palm including all the five fingers, not only the thumb.

2.2 Therapy

Of all the parts, *Therapy* is our one of the most important and focused topic. For stroke patients with upper limb disability, rehabilitation comes with routine wise therapy. There are four basic exercises for stroke patients with upper limb disability, or people whose hands are too tight to move or the hand movement is limited.[1] The basic exercises are:

1. Make a full fist
2. Spread your fingers
3. Bring the thumb to each finger tip
4. Round your hand for a functional “C”



(a) Fisting Exercise

(b) Finger Spreading Exercise

(a) Pinching Exercise

(b) Griping Exercise

Figure 2.2: 4 Basic Exercise for rehabilitation for Upper Limb Disability [1]

From these 4 exercises, we are only focused on the *pinching* exercise (figure 2.2a) in this thesis.

2.3 Dataset

As previously discussed, much work has been done on human-robot interaction based rehabilitation for upper limb disability based patients. That is why, many datasets are available. The HANDS dataset was produced for human-robot interaction research and is made up of RGB and Depth frames that are geographically and temporally matched. It includes 12 static single-hand gestures performed with both the right and left hands, as well as 3 static two-hand gestures, for a total of 29 distinct classes.[9] But as we are focused on pinching exercise, the dataset will not be very helpful for us. But it gave us insight on how to properly annotate data to do research tasks like this. The HANDS dataset also focuses on human-robot based interaction, which is not used in our work, but it suggests a procedure to properly collect and annotate data.

Another dataset we found that was similar to what we required was the putEMG dataset [10]. Although this is a well annotated dataset, the problem that we discovered is that this dataset uses *sEMG*, which is known as surface electromyography. Electromyography is a kind of signal which includes recording muscle activity using on-skin electrodes. A image given below shows proper representation of sEMG used in up given case.

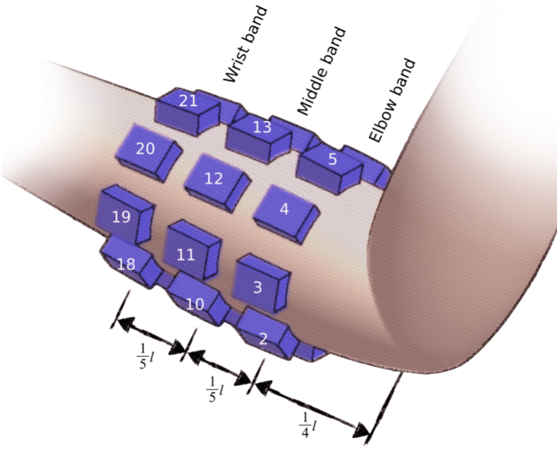


Figure 2.3: Electrode placement and numbering [10]

But as we are focused on vision based approach, we were focusing on basic image data, not sEMG. This dataset gave us insight on what type of images to collect and in what environment and background conditions these images should be collected in.

We also found a dataset from kaggle with the title *Hand Gesture Recognition Database*[11] which could have served our purpose but it was not used due to the fact that it did not help us detect the transition of the exercise from the start state to finish state as we needed. If we use video feed, we will get a transition state in between start and finish. Which we are calling the *Middle* state. It plays a vital role in our proposed approach because if a stroke patient cannot touch the finish state in pinching exercise, he may have come covering certain distance from the *home* state which may be called improvement by our result metric. It is a important factor because we do not want our user to use any kind of wearable.

2.4 Existing Approaches

For our proposed approach, we have gone through some references to find the suitable one which served our purpose. Firstly we went with the suggested method of using *Bag of Words*. For almost a decade, the Bag of Words (BoW) paradigm has been utilized in machine vision. When it comes to machine vision applications, the model has gained popularity due to its simplicity and efficacy (Li, Dong, Xiao,

& Zhou, 2016), and it is also known as the bag of visual words and the bag of features. The technique extracts visual words from training images to generate the dictionary, also known as a codebook, which comprises of visual words, as represented by the flowchart in figure below.[12]

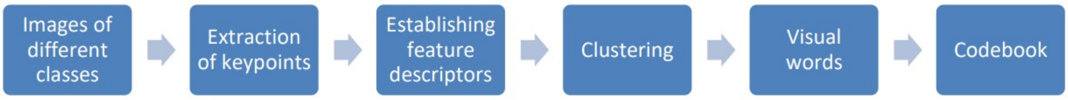


Figure 2.4: BOW - flow of work[12]

During the learning stage, a huge collection of photos from various classes is employed. The extraction of keypoints from each image is the first step. Following that, for each keypoint, feature descriptors that represent the keypoint's surroundings are established. Following that, for dimension reduction reasons, these descriptors are organized into groups known as visual words. The codebook, which is akin to a dictionary containing the lexicon of words, collects all of the created visual words from the training images. [Textbox - why we did not use bag of words]

Almost completely automated rehabilitation method based on computer vision focuses on stroke survivors and gait. As a result, this project was created to help with hand injuries by creating a computerized hand deviation exercise. This exercise was also beneficial to patients suffering from hand injuries such as carpal tunnel syndrome, tendon discomfort, and climber's elbow. Ulnar-radial movement is a very common and popular exercise for those who have hand or wrist problem suggesting upper limb disability. Wrist extension and flexion, hand / finger tendon glide, and wrist supination / pronation are other exercises that are beneficial for injured hands. [3]. The Figure suggest how ulnar-radial movement.

But in this paper, we are focusing on the finger movement and coordination(Pinch Exercise). But this research work indicates a great strategy about computer vision based tele-rehabilitation process. We focused on copping with same kind of strategy for our proposed approach.

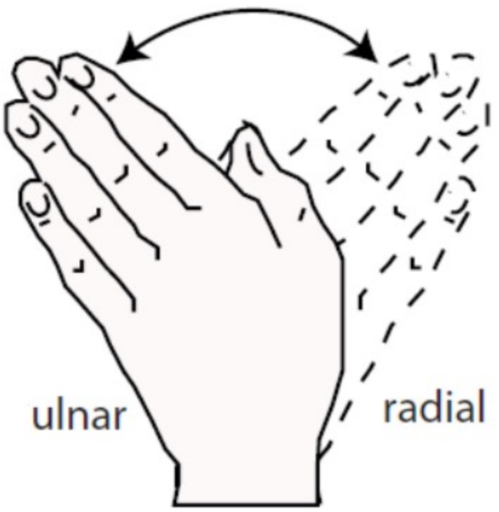


Figure 2.5: Ulnar and radial movement therapy[3]

3 Proposed Methodology

The approaches we took for making our system can be divided into two segments:

- Collecting images of therapeutic hand gestures to build the data-set.
- Building and training a deep learning model to identify and recognize the gestures.

We will discuss the methodologies for each segment in the following sections.

3.1 Dataset Generation

3.1.1 Overview of Data Collection Process

For our data collection process, we had to manually take photos of different hand postures from our participants. Among all the students and teachers of our university, 47 participants were randomly chosen. The participants were a near even distribution of males and females. Our data collection process was carried out in one of the labs of our university, Islamic University of Technology (IUT). The setup included a computer that had two types of webcams mounted on the monitor, a high-resolution one and a low-resolution one. The low-resolution camera was a 720p camera and the high-resolution one was a 1440p camera. Participants were seated in front of the cameras and were given instructions on specific hand gestures. The participants would perform the specific hand gestures and we would take images of them using both the cameras, one camera at a time. The images would automatically be annotated using the interface (PICS) that we built, and be saved in the correct folder. The images taken using the high-resolution camera were then resized to match the size of the images taken using the low-res camera.

3.1.2 Experiment Design

Experiment design is the process of deciding what variables to use, what procedures to use to conduct the experiment and gather data, and how many participants to use, among other things. For our experiment, we had five independent variables: *camera used, hand used, pose, distance from camera* and *perspective angle from camera.* The variables are constructed in a tree-like hierarchy.

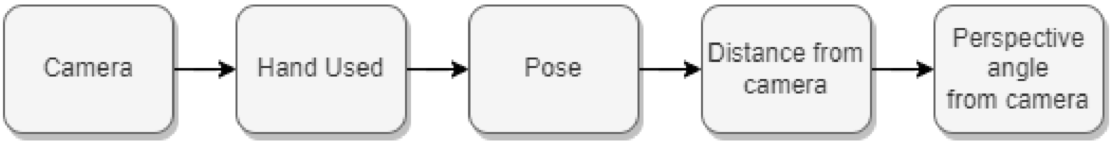
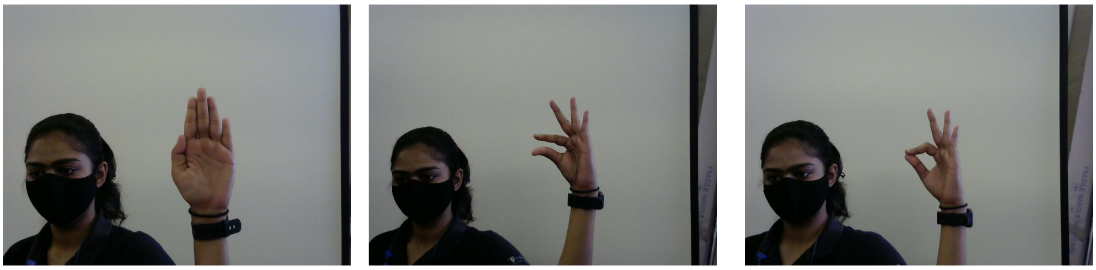


Figure 3.1: Experiment Design Hierarchy

This can be better understood from figure 3.3. What this means is that, we are taking unique images for each camera-hand-pose-distance-angle combination. With 5 images for each unique posture, a total of **540** images was taken from each participant. All of these images were saved in a hierarchical folder structure and annotated based on the unique gesture.



(a) Home gesture (b) Middle gesture (c) Action gesture

Figure 3.2: The 3 types of gestures

We have selected 3 gesture poses for our dataset. Figure 3.2a is the *Home* gesture, which is the initial position of the hand. Figure 3.2b is the *Middle* gesture, which is the transition state between the home and action gesture. Figure 3.2c is the *Action* gesture, which is the final position and the accurate pinching position of the hand. We have included the middle gesture as a transition between the home and action gestures, and this will make our deep-learning models more robust.

We used three values for the `distance` independent variable in our experiment. *Near* distance referred to distance between 10-30 cm of the camera, *medium* distance referred to distance between 30-50 cm of the camera and *far* distance referred to distances between 50-70 cm of the camera. These distances were strictly maintained during the experiment.

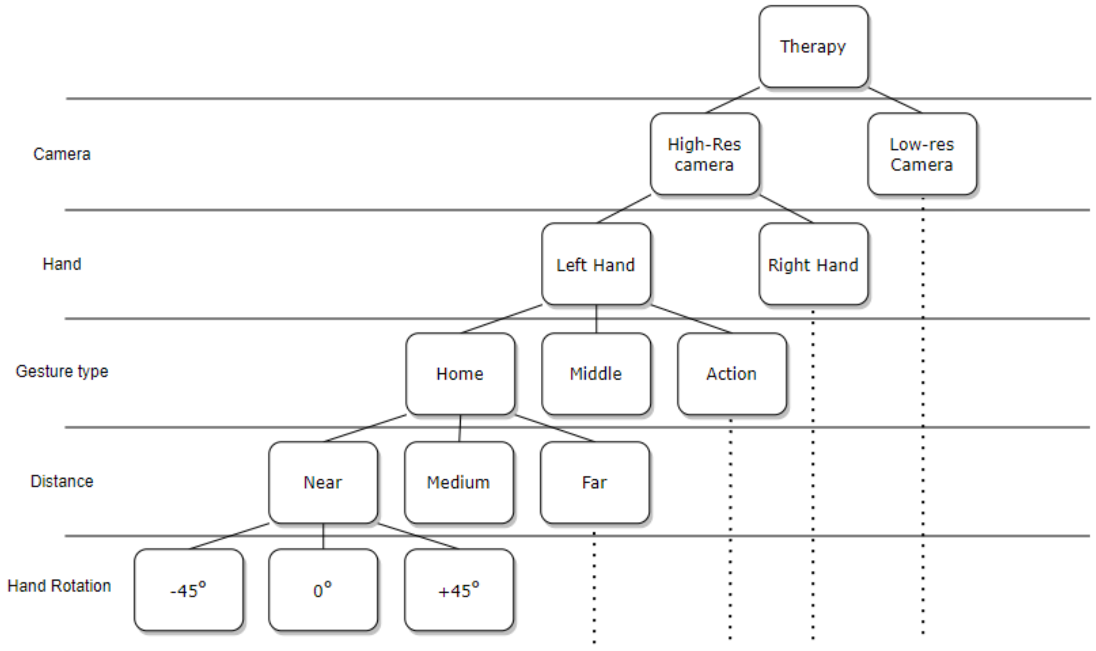


Figure 3.3: Tree structure for image collection

We had one control variable in our experiment, which was the lighting condition. We wanted to have two different lighting conditions, bright and dark. But due to environmental restrictions we could not achieve desired results with the dark condition. Both the conditions provided very similar images and we could not clearly differentiate between the two. That is why, we changed this to a control variable and kept the lighting condition to always be bright.

We had a few random variables in our experiment which did not contribute to the outcome of the experiment. Things such as participant age and gender did not affect the output of our experiment in any way, but we could not control these variables. So these remained in our experiment as random variables.

So to conclude, we had the following variables in our experiment:

- **Independent variables**: Camera used, hand used, pose, distance from camera and perspective angle from camera
- **Control variables**: Lighting condition
- **Random variables**: Participant age, gender etc.

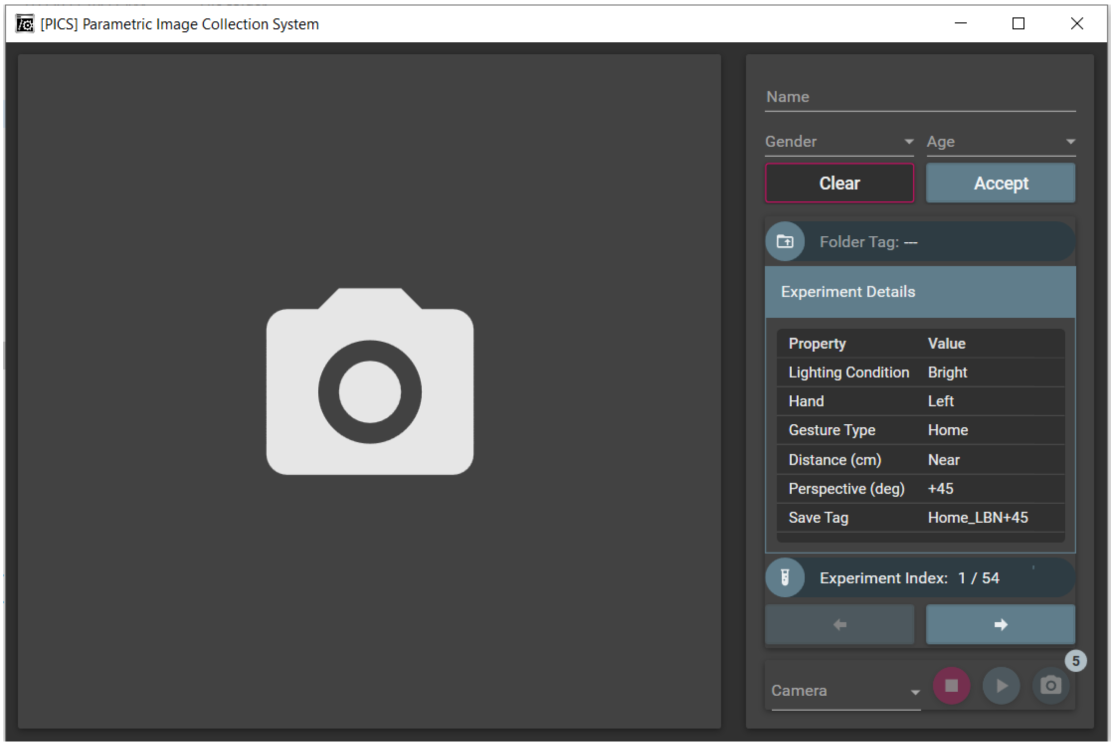
3.1.3 Interface for automatic annotation (PICS)

Annotating data-sets can be a very redundant and time-consuming task. To automatically annotate our data-set and save them in the correct hierarchical folder, we created an interface software called *Parametric Image Collection System* (PICS). This software allowed us to take images using multiple different webcams (two in our case) and automatically annotate images and save them in the correct hierarchical folder. The experiment details can be changed using the `.json` file in the system. The camera window will display the video feed from the webcams and we can click on the little camera button on the bottom right corner to take a snapshot of the current video feed. Figure 3.4 is the interface for PICS. The different types of camera inputs can be selected from the drop-down at the bottom.

For automatic annotation, the naming convention we followed is as follows:

```
GestureName_HandLightDistanceOrientation_Serial
```

An example of a the name is `Action_LBF+45_1`. Here `Action` refers to the type of gesture, `L` denotes Left hand, `B` denotes bright condition, `F` denotes Far distance, `+45` denotes the orientation and `_1` denotes the serial number of the image. The images are kept in separate folders according to the type of camera used. The system was created in .NET framework.



[PICS] Parametric Image Collection System

Folder Tag: —

Experiment Details

Property	Value
Lighting Condition	Bright
Hand	Left
Gesture Type	Home
Distance (cm)	Near
Perspective (deg)	+45
Save Tag	Home_LBN+45

Experiment Index: 1 / 54

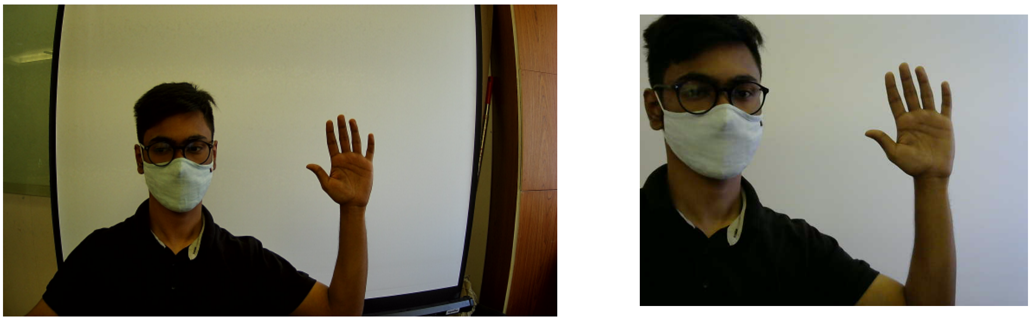
Figure 3.4: PICS automatic annotation interface

3.1.4 Two types of cameras

We used two types of cameras for taking the image:

1. Logitech C270 HD Webcam
2. Fantech Luminous C30 2K Webcam

The first camera is a 720p HD webcam, which is the low-resolution webcam in our context. The second camera is a 1440p webcam, which is the high-resolution camera in our context. We used two types of cameras to include more variations on our dataset. The two camera outputs are very different, because they have different resolutions and field-of-view (FOV). The high-resolution camera has a wider FOV, while the low-resolution camera has a normal FOV.



(a) Image from Fantech C30

(b) Image from Logitech C270

Figure 3.5: Comparison between the different types of camera outputs

Figure 3.5 shows us the differences between the images taken by the two types of cameras. Figure 3.5a is a wide-angle picture and has a lightly warmer color signature. While on the other hand, 3.5b is a has a normal FOV and has a slightly cooler color signature. By introducing two types of cameras, we managed to introduce greater variety in the data-set so that models trained using our data-set will be robust.

3.2 Building and training a deep learning model

The next segment of our work focuses on training a machine learning model that will be able to accurately predict the therapeutic gesture that is being performed from a live video-feed and also be able to measure the accuracy of the gesture. For measuring the accuracy, we would need to come up with a metric system that would properly measure the accuracy. We will not be looking into the measurement metric for now, and will focus on the model building and model training. The metric generation will be included in the future work.

3.2.1 Overview of model building

We trained three types of models for our system. The three types of models are described below:

3.2.1.1 Model trained on hand-landmarks

- **Model Overview**: The first model we trained was trained on the `.csv` data that we plotted from the hand landmarks using Google's Mediapipe Hands model. The Mediapipe Hands is able to identify the hand in an image and places 21 landmark markers on the hand. The model is a combination of two models. The first model is a palm recognition model that works on the entire image and returns a bounding box of the hand region. The second model works on the cropped hand region inside the bounding box to detect the hand and generate the hand landmarks. The hand landmarks are given in figure 3.6.

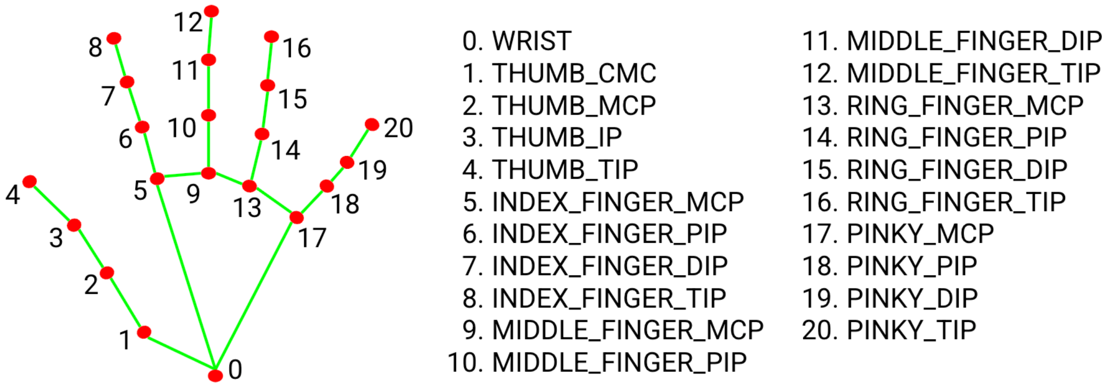


Figure 3.6: Hand landmarks generated by Mediapipe Hands

We trained a deep learning CNN model that works on the landmark coordinates. The model architecture consists of very simple feed-forward neural networks. All of the hidden layers are fully connected dense layers and the output layer is also a dense layer with the *softmax* activation function. The model architecture is given in figure 3.7.

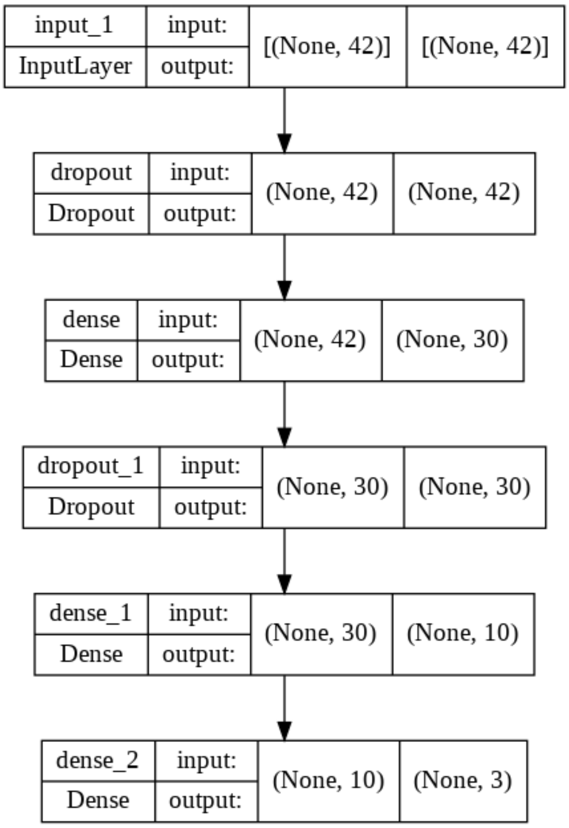


Figure 3.7: Model architecture for the hand landmarks model

From figure 3.7, we can see that the input layer is a Dense layer with the shape of 42. This is because we have 42 landmark coordinates in the `.csv` file that we are using as input to the neural network. Then we have a *droput* layer that reduces overfiting of the model by randomly setting some input values to 0. Following the dropout layer, we have two more pairs of Dense and dropout layers. Then we have a dense layer that feeds to the final output layer. All of the above layers have the *relu* activation function. The final layer has the *softmax* activation function. This function has 3 neurons, which means there are 3 outputs available. The *softmax* functions calculates the possibility of each output class and returns the output class with the maximum possibility.

- **Data-preprocessing for model training**: Our data preprocessing consists of plotting the coordinate data points in the `.csv` file. We first processed each image through the Mediapipe Hands model to detect the hand landmarks. Then, for each image, the X and Y-coordinates for each generated landmarks were found out and plotted into a single row of the `.csv` file. The label for each image was also plotted at the last column of the file. We fit this generated `.csv` file into our machine learning model architecture and trained our model.

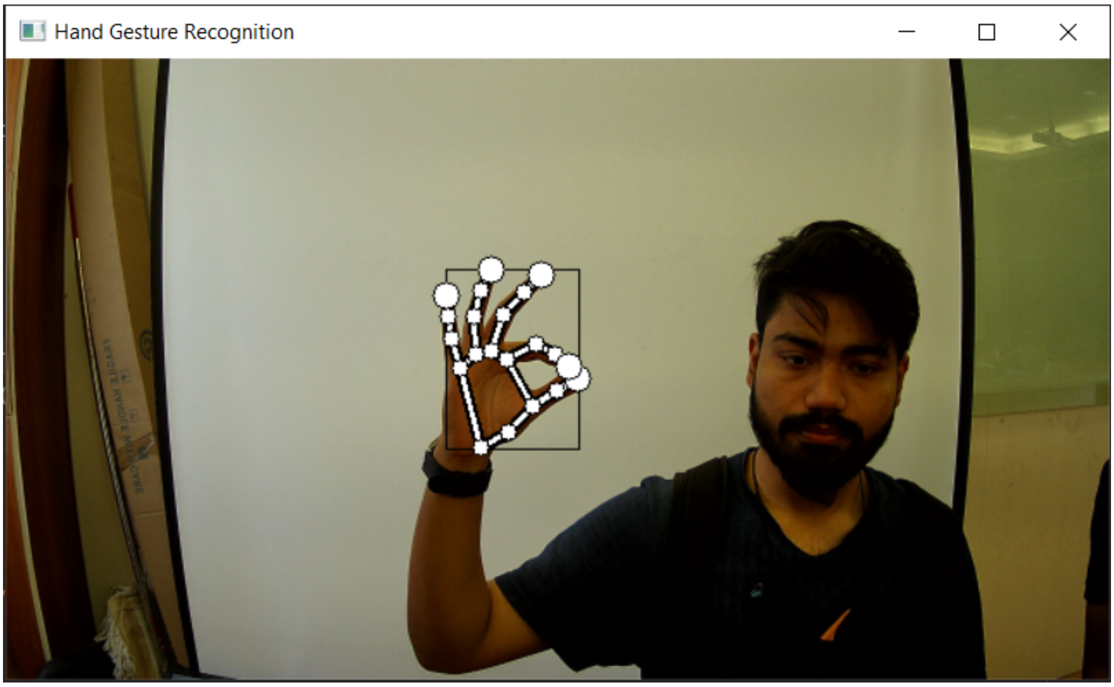


Figure 3.8: Hand landmarks on training image

3.2.1.2 Image Classifier Model

- **Model Overview**: The second model we trained is a image classifier model. The model works on raw image input and gives us the correct label as the output. The model performs convulations on the image pixels and automatically extracts the features from the images and learns from those features.

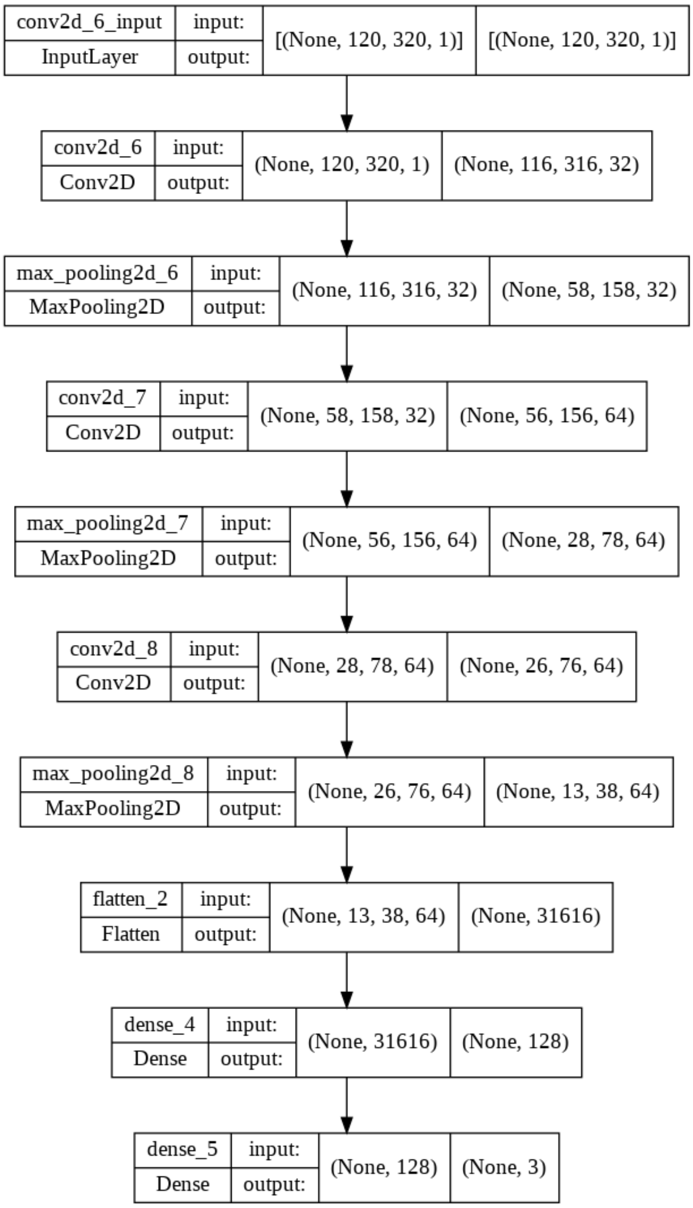

conv2d_6_input | input: | [(None, 120, 320, 1)] | [(None, 120, 320, 1)]
InputLayer | output:
conv2d_6 | input: | (None, 120, 320, 1) | (None, 116, 316, 32)
Conv2D | output:
max_pooling2d_6 | input: | (None, 116, 316, 32) | (None, 58, 158, 32)
MaxPooling2D | output:
conv2d_7 | input: | (None, 58, 158, 32) | (None, 56, 156, 64)
Conv2D | output:
max_pooling2d_7 | input: | (None, 56, 156, 64) | (None, 28, 78, 64)
MaxPooling2D | output:
conv2d_8 | input: | (None, 28, 78, 64) | (None, 26, 76, 64)
Conv2D | output:
max_pooling2d_8 | input: | (None, 26, 76, 64) | (None, 13, 38, 64)
MaxPooling2D | output:
flatten_2 | input: | (None, 13, 38, 64) | (None, 31616)
Flatten | output:
dense_4 | input: | (None, 31616) | (None, 128)
Dense | output:
dense_5 | input: | (None, 128) | (None, 3)
Dense | output:

Figure 3.9: Model architecture for the image classifier model

From figure 3.9, we can see that the input layer is a *Conv2D* layer that has the shape of 120 by 320. This is the size of the image that we are going to input into the CNN. The input layer is followed by another *Conv2D* layer with the same shape. Then we have a *MaxPooling2D* layer that downscales the output from the Conv2D layer by using the maximum value from the input window over each input channel. Then we have 2 more pairs of Conv2D and MaxPooling2D layers. Then we have a flatten layer that changes the shape of the data from multi-dimensions to a single dimension. After that we have two fully connected dense layers that feeds into the final output layer. All of the above layers use the *relu* activation function. The final output layer is a dense layer that has the *softmax* activation function.

- **Data pre-processing for model training**: The first step in data pre-processing was to remove the bad images. There were some images which were not properly distinguishable, so we had to remove those images. The second step was to change the color of the images and resize them. Since we used the *OpenCV* library, we had to change the color of each image before we could use it in processing. Next, we had to resize the images to a size that can be easily processed by the CNN model. We resized each image to a 16 by 16 pixel size. Then each model was fitted to the CNN model. The CNN model was automatically able to extract features from the images and train itself.

3.2.1.3 Ensembled Model In machine learning, ensembling is a method of uses the outputs of several base models to create a model that will perform better than all of the individual models. Our model trained on hand landmarks did not provide good enough accuracy results. The image classifier was able to show accurate results, but performed poorly when it came to recognizing gestures from real-time video feed. Both the models provided similar results in this regards. Since the results were not satisfactory, we wanted to create a new model that was made from the combination of the two previous models.

Ensemble Process Overview: The ensembled model was made from averaging the per class prediction of the two previous models. Since both the previous models used the *softmax* activation function and both of them had 3 output classes, so the models gave a probability on what it thinks is the correct class for the given input. We took these per-class prediction probabilities made by the two previous models and made an average of them. Using the average probabilities, we measured the accuracy of the model against the same set of inputs. As we will discuss in the results section, we managed to achieve slightly better results with the ensembled model then we did with the previous two models.

4 Results & Discussion

In this section, we will discuss the different evaluation metrics of our three models and provide a contrast between them.

4.1 Evaluation Metrics

The evaluation metrics that we considered for our experiments are:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$F1score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{3}$$

Here, TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

For the labels:

- Label '0' denotes the `Home` position.
- Label '1' denotes the `Middle` position.
- Label '2 denotes the `Action` position.

4.2 Result Analysis

In this section we will discuss the performance of each of the models that we trained. The classification reports of each of the model is given one after another.

Classification Report (Landmark Model)				
	precision	**recall**	**f1-score**	**support**
0	1.00	0.95	0.97	609
1	0.85	0.30	0.44	639
2	0.61	0.99	0.75	694
accuracy			0.75	1942
macro avg	0.82	0.75	0.72	1942
weighted avg	0.81	0.75	0.72	1942

Table 4.1: Classification report of the landmark model

Table 4.1 shows us that the accuracy of the first model is 75%. The model performs well on the '0' and '2' labels, but performs very poorly on the '1' label, achieving a f1score of only 44%.

Classification Report(Image Classifier)				
	precision	**recall**	**f1-score**	**support**
0	1.00	0.98	0.97	609
1	0.98	0.96	0.96	639
2	0.97	0.99	0.96	694
accuracy			0.96	1942
macro avg	0.98	0.97	0.96	1942
weighted avg	0.97	0.97	0.97	1942

Table 4.2: Classification report of the Image Classifier model

Table 4.2 shows us that the accuracy of the first model is 96%. The model performs very well on all the labels, achieving very similar f1scores on all of the labels. Although this models performs very well for the test images, when we used it for gesture recognition from a real-time video feed, it did not perform very well.

Classification Report(Ensemble Model)				
	precision	**recall**	**f1-score**	**support**
0	1.00	0.96	0.97	609
1	0.92	0.63	0.71	639
2	0.80	0.99	0.85	694
accuracy			0.84	1942
macro avg	0.90	0.85	0.84	1942
weighted avg	0.89	0.86	0.85	1942

Table 4.3: Classification report of the ensemble model

Table 4.3 shows us the classification report for the ensemble model. The ensemble model has an average accuracy score of 84%. From the f1scores, we can see that the model performs well for the ’0’ and ’2’ labels, but it does not perform well for the ’1’ label.

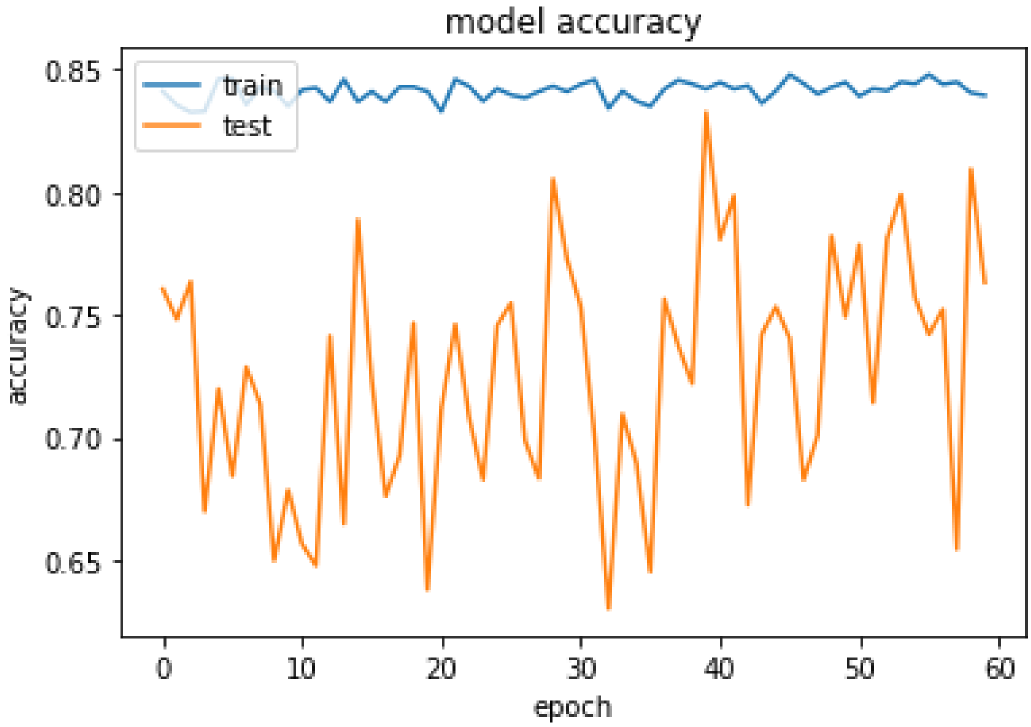


Figure 4.1: Accuracy for landmarks model

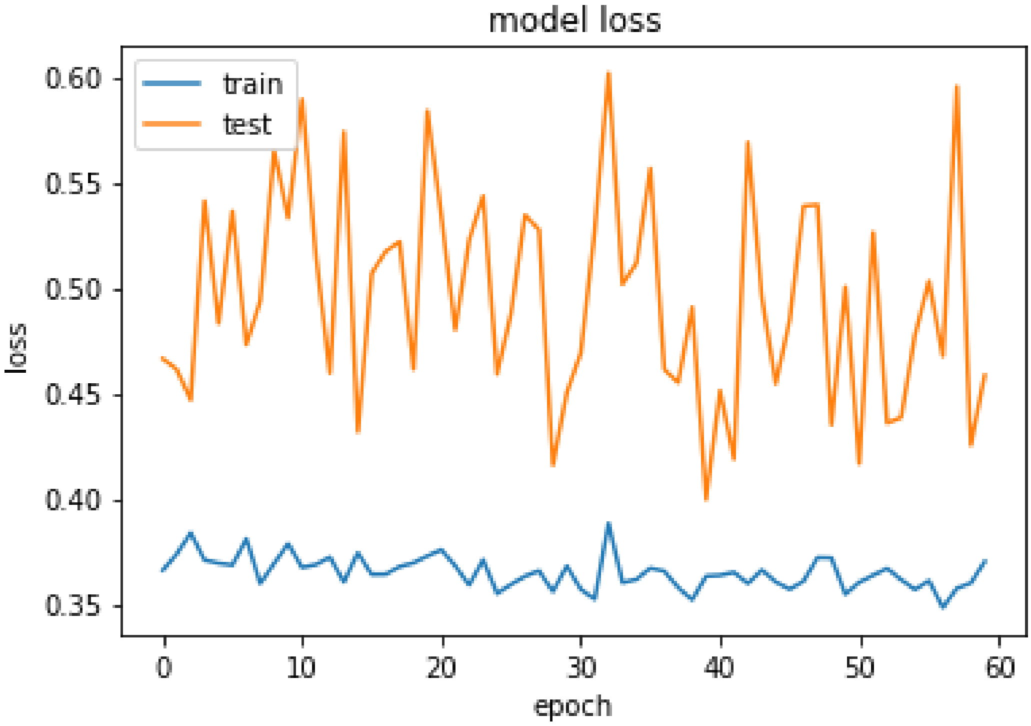
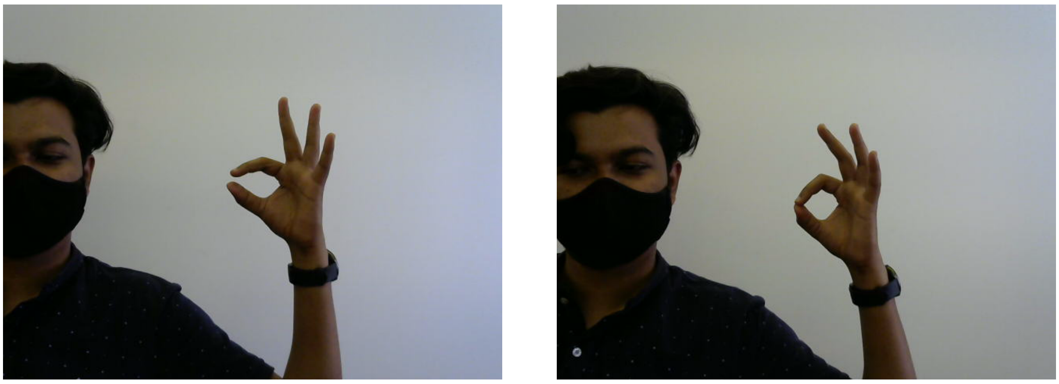


Figure 4.2: Loss for landmarks model

From figures 4.1 and figure 4.2, we can see that the hand landmarks model fluctuates a lot on the test data.

Reasons for fluctuation

The hand landmarks model fluctuates a lot on test data due to the '1' and '2' labels, or more precisely, the `Middle` and the `Action` gestures. The two gestures are very similar, and the model sometimes classifies one gesture as the other. The `Home` gesture is quite different from the rest of the gestures and the model accurately predicts that gesture, but it has trouble with the `Middle` and `Action` gestures. This is because CNN models tend to converge to the nearest known class, if an input is given between two classes. That is why the models tends to converge towards the wrong class sometimes. Also, the training data contains some gestures that are actually the `Middle` gesture, but is very close to the `Home` gesture. This can also be the reason why the model did not learn properly on what type of class the gesture belongs to. From figure 4.3, we can see how close the two gestures are.



(a) Middle gesture

(b) Action gesture

Figure 4.3: Comparison between Action and Middle gestures

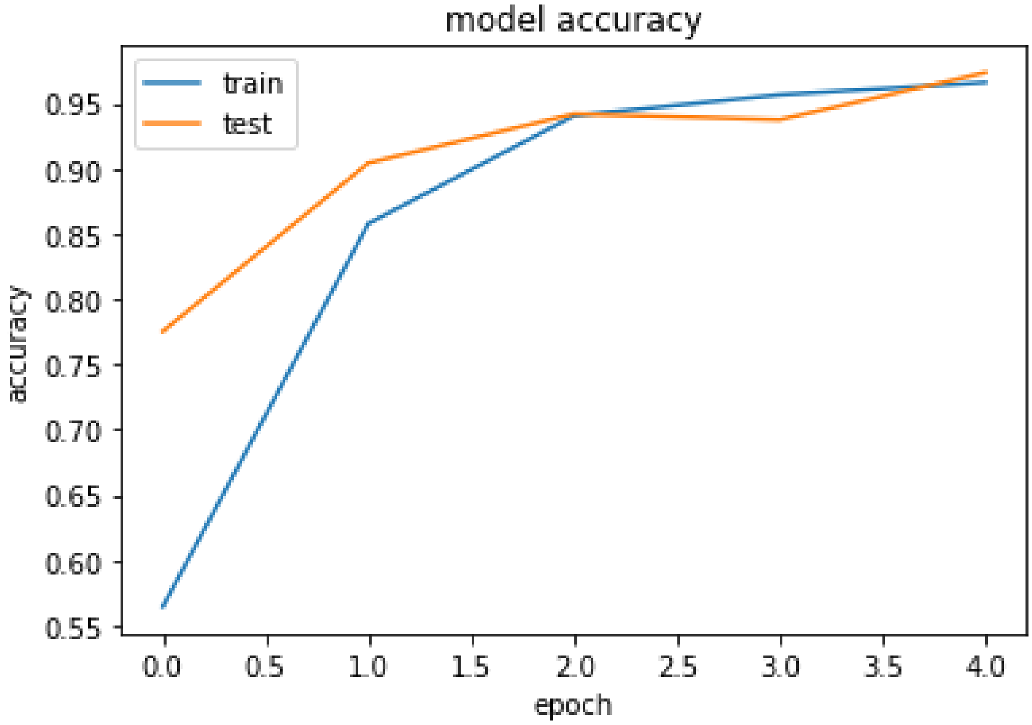


Figure 4.4: Accuracy for Image classifier model

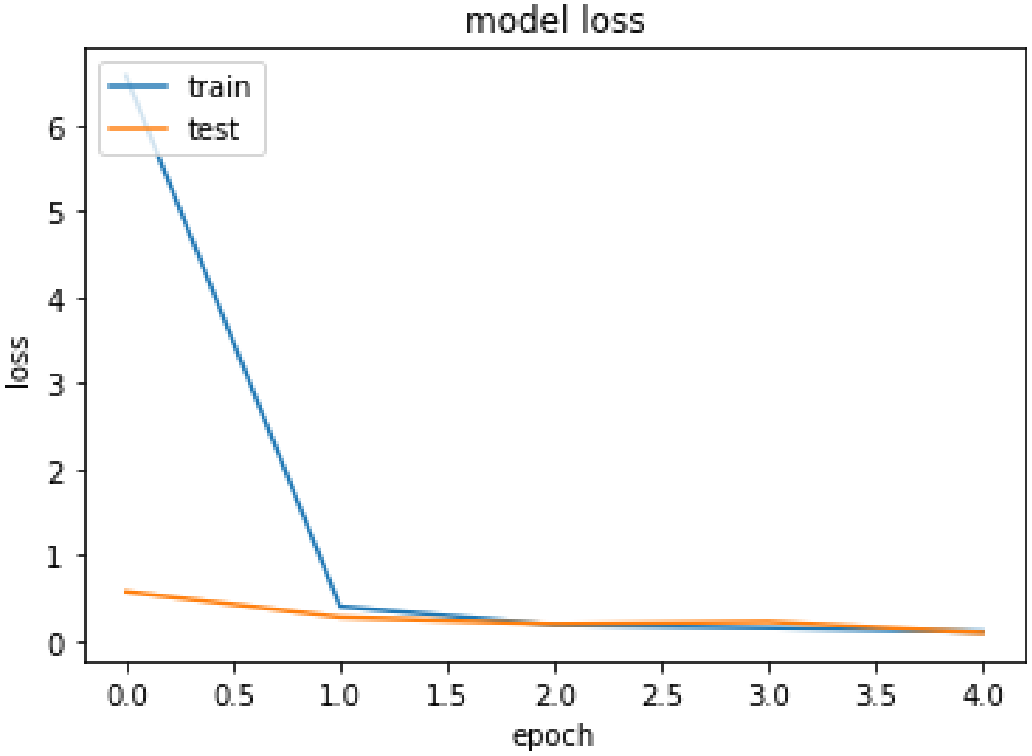


Figure 4.5: Loss for Image classifier model

Figure 4.4 and figure 4.5 shows us the accuracy and loss for the image classifier model. Although the model performed very well on the test sets, it did not perform well on the real-time video feed. That is why we opted for the ensemble model.

4.3 Image classification by the hand landmarks model

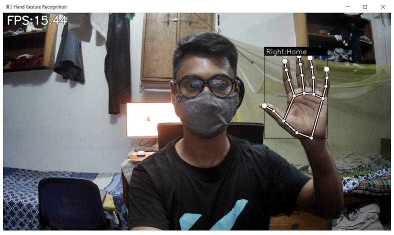


Figure 4.6: Home Gesture

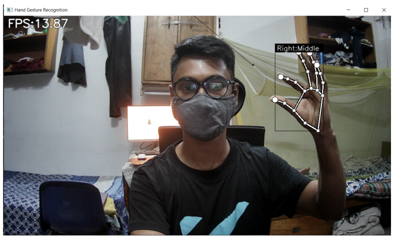


Figure 4.7: Middle Gesture

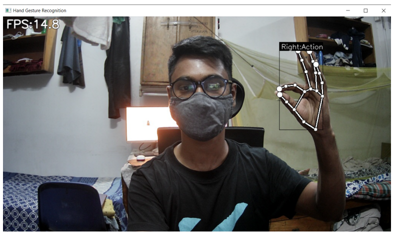


Figure 4.8: Action Gesture

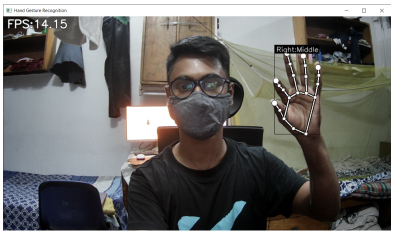


Figure 4.9: Middle Gesture Sideways

The four images show us that the hand landmarks model is capable or correctly predicting the hand gesture from a real-time video feed. For a real-time video feed, each frame of video is analyzed and the landmarks are extracted from them. Then the model is used to recognize what type of gesture the performed action is.

4.4 Image classification by the image classifier

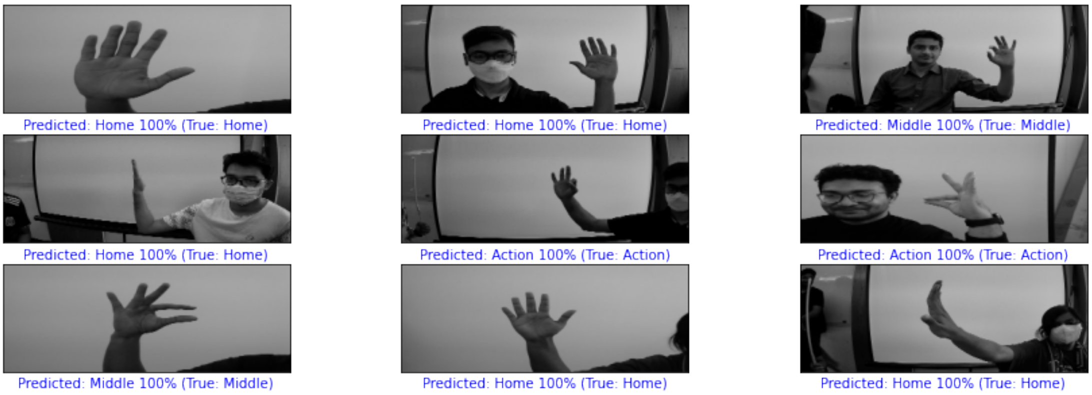

Predicted: Home 100% (True: Home)
Predicted: Home 100% (True: Home)
Predicted: Middle 100% (True: Middle)
Predicted: Home 100% (True: Home)
Predicted: Action 100% (True: Action)
Predicted: Action 100% (True: Action)
Predicted: Middle 100% (True: Middle)
Predicted: Home 100% (True: Home)
Predicted: Home 100% (True: Home)

Figure 4.10: Image classified using the image classifier model

Figure 4.10 shows us the output produced by the image classifier model. The image classifier model is correctly able to predict the gestures based on the test images. Here we have plotted 9 images, of which all of them have been correctly predicted by the model.

4.5 Weakness of our models

4.5.1 Hand landmarks model

The weakness of our hand landmarks model is that the accuracy is not very good. It has only abut 75% accuracy. It also cannot differentiate very well between the two similar gestures, `Middle` and `Action` gestures.

4.5.2 Image classifier model

The weakness of the image classifier model is that is does not perform very well on the real time video feed images. Even though it performed extremely well on the test dataset, it failed to perform well on the real time video feed. Since our end goal was to classify videos from a real-time video feed, it still has weaknesses.

4.5.3 Ensemble Model

The ensemble model performs well on the test sets, but not as good as the image classifier model. We are yet to test it on real-time video feed.

5 Conclusion

5.1 Summary

To conclude our work, we have collected and built an extensive dataset for the pinching hand exercise. Alongside that, we have used 3 deep learning models to detect therapeutic hand gestures, 2 of them have been tested on real-time video feed. This will later on help the patients with upper limb disability by doing exercises from home. To be precise, we wish to ensure tele-rehabilitation so that life of those who requires physio-therapy on a regular basis gets easier.

5.2 Future Work

For future work, we wish to perform the following tasks:

- Improve the accuracy of our model.
- We would also like to focus on training and testing light-weight models so that the system can be imported to Internet of Things (IoT) devices.
- We will develop an evaluation metric that we will use to predict the accuracy of the therapeutic hand gesture that is being performed from a live video feed.
- Our current dataset consists of only pinching exercise images. We would like to extend this dataset by including images of other exercises as well.

References

[1] Easy Hand Exercises to Boost Recovery from a Stroke
shorturl.at/npG79

[2] A Gentle Introduction to Ensemble Learning Algorithms
https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/

[3] Bakar, M. Zabri Abu, et al. ”Computer vision-based hand deviation exercise for rehabilitation.” 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE). IEEE, 2015.

[4] Decker, Jilyan, et al. ”Wiihabilitation: rehabilitation of wrist flexion and extension using a wiimote-based game system.” Governor's School of Engineering and Technology Research Journal (2009): 92-98.

[5] Strong, Kathleen, Colin Mathers, and Ruth Bonita. ”Preventing stroke: saving lives around the world.” The Lancet Neurology 6.2 (2007): 182-187.

[6] Langhorne, Peter, Julie Bernhardt, and Gert Kwakkel. ”Stroke rehabilitation.” The Lancet 377.9778 (2011): 1693-1702.

[7] Carpinella, Ilaria, Johanna Jonsdottir, and Maurizio Ferrarin. ”Multi-finger coordination in healthy subjects and stroke patients: a mathematical modelling approach.” Journal of neuroengineering and rehabilitation 8.1 (2011): 1-20.

[8] Pompili, Giulia, et al. ”Development of a Low-cost Glove for Thumb Rehabilitation: Design and Evaluation.” 2020 IEEE International Conference on Human-Machine Systems (ICHMS). IEEE, 2020.

[9] Nuzzi, Cristina, et al. ”HANDS: an RGB-D dataset of static hand-gestures for human-robot interaction.” Data in Brief 35 (2021): 106791

[10] Kaczmarek, Piotr, Tomasz Mańkowski, and Jakub Tomczyński. ”putEMG—a surface electromyography hand gesture recognition dataset.” Sensors 19.16 (2019): 3548.

[11] Hand Gesture Recognition Database
`https://www.kaggle.com/datasets/gti-upm/leapgestrecog`

[12] Qurratu'aini, Dayang, et al. "Visual-based fingertip detection for hand rehabilitation." Indonesian Journal of Electrical Engineering and Computer Science 9.2 (2018): 474-480.

[13] Zhang, Yin, Rong Jin, and Zhi-Hua Zhou. "Understanding bag-of-words model: a statistical framework." International Journal of Machine Learning and Cybernetics 1.1 (2010): 43-52.

[14] Lei, Xinyu, Hongguang Pan, and Xiangdong Huang. "A dilated CNN model for image classification." IEEE Access 7 (2019): 124087-124095.

[15] Agarap, Abien Fred. "An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for image classification." arXiv preprint arXiv:1712.03541 (2017).

[16] Helpful Hand Exercises for Stroke Patients of All Ability Levels
`https://www.flintrehab.com/hand-exercises-for-stroke-patients/`

[17] Stroke (Cerebral Vascular Accident (CVA) and Spinal Stroke)
`https://www.christopherreeve.org/living-with-paralysis/health/causes-of-paralysis/stroke`