

# **Implementation of a UI-Based Hand Gesture Interaction System for Character Input and Subsequent Action Execution**

**\*Chintam Sravan Kumar, \*Mohamed Iqbal**

## **1.ABSTRACT**

Gesture-based UI Interaction is a process that facilitates user-system interaction without the need for physical touching. This method allows users to input characters or numbers and execute corresponding actions through gestures. The system displays symbols representing specific characters or numbers, and by presenting a particular action symbol, the associated action is performed. Particularly during viral crises, there is a considerable demand for touchless systems that prioritize minimal physical contact, thereby reducing transmission risks, promoting hygiene practices, and upholding public confidence. The implementation of touchless systems during viral crises involves the utilization of a Convolutional Neural Network (CNN) algorithm to detect hand gestures performed by the user. The CNN effectively identifies the hand gestures and translates them into the appropriate characters or executes the required actions. In the training phase, the accuracy achieved by the alphabet dataset is 0.912, the numeric dataset is 0.922, and the actions dataset is 0.933, while in the testing phase, it reaches to 0.924 for the alphabet dataset, 0.941 for the numeric dataset and 0.953 for actions dataset. The combined accuracy achieved by Alphabet, Numeric and Action dataset is 0.922 in the training phase and 0.939 in testing phase.

### **KEYWORDS:**

- CNN
- Hand Gesture
- UI-Interaction
- Contactless Interaction
- Image Recognition

## **2. INTRODUCTION**

There is a huge need for a service that makes contactless interaction with a device because of some viruses which do not allow us to interact with devices by touching them. There is an increase in diseases that come through contacting other human beings or devices which are used by them. There is a high need for a device that can do all its processes or transaction by taking input without any contact with others. So, GUI Based Interaction can help in doing contactless transactions with the hand gestures of human beings. This proposed work allows users to input commands using hand gestures, and it performs further actions based on the user's needs. This touchless interaction is particularly useful during situations when physical contact should be avoided, such as during a pandemic like COVID-19.

Few gesture-based applications are utilized in various fields. One such application is Sign Language Translation, where hand gesture-based UI design can effectively translate sign language into spoken or written language in real-time[10]. Another prominent domain for gesture-based applications is Gaming, where they have gained popularity, especially with the use of motion-sensing controllers like the Microsoft Kinect or the Leap Motion Controller[11].

Furthermore, gesture-based applications find significant utility in Human-Robot Interaction, as they facilitate seamless communication and interaction between humans and robots. Moreover, the integration of hand gesture-based UI design can be observed in wearable devices, such as smartwatches or fitness trackers. In these devices, advanced deep learning-based algorithms are employed to detect specific hand gestures, enabling users to execute various actions on the respective UI design effectively.

One of the deep learning algorithms CNN (Convolution Neural Network) is frequently utilized in so many researches based on image detection. The proposed study employs the Convolutional Neural Network (CNN) algorithm for the purpose of detecting and recognizing the displayed hand gestures.

A CNN may be a kind of arranged engineering for profound learning calculations and is particularly utilized for picture acknowledgment and assignments that include the preparation of pixel data. This makes them profoundly appropriate for computer vision (CV) assignments and for applications where question acknowledgment is crucial, such as self-driving cars and facial acknowledgment. There are many applications of hand gestures but the GUI-based Interaction is not available. It is very much necessary for an Interaction to be accomplished without any touching. Also, it increases the comfort of handling the interaction between a system and a human.

### **3. RELATED WORK**

In recent years, numerous researchers have conducted investigations on gesture-based analysis. Ashutosh Gupta, Yogesh Kumar, and Sanal Malhotra [1] proposed framework is utilized for keeping money secure and can be utilized for companies or at personally secured places for security purposes. They have proposed a security-based motion acknowledgment system using a web camera. They have utilized skin discovery and canny location calculation to distinguish a motion from a picture. The authors of [2] will utilize the camera for motion detection, while a non-vision-based technique will involve the use of sensors. In this consideration, a vision-based strategy was utilized. This contraption distinguishes and finds hand developments in orchestration to keep a communication channel open with others. Utilizing convolutional neural frameworks and made neural frameworks, this asks almost makes a flag affirmation system. This consideration looks into the inclinations and obstructions of hand development affirmation. In [3] they created a dataset and after that utilized a lightweight CNN that appears to recognize and classify hand improvements capably. They additionally assessed the classification precision with a compelled number of diagrams in a video movement. They compare the significant camera's video flag affirmation execution to that of the RGB camera. They evaluate the proposed model's execution on edge computing contraptions and compare it to benchmark models in terms of precision and derivation time. The authors [4] aims to assist people with disabilities in using American Sign Language and spoken language through their research. Deaf people will be able to easily communicate or interact with other people. This study proposes gestures or signs to search

for learned words with the YOLOv3 algorithm, which aims to identify gestures or languages that recognize that their letters are equal. The authors [5] tried to efficiently and intelligently obtain the same information as the background in the skin field, this study presents a model based on the development of the centroid basin algorithm (ICWA) and dual-channel convolutional neural network (DCCNN). The effectiveness of this approach comes from using ICWA to segment the raw image accurately. The author [6] presented a real-time navigation system based on the TSL (colour, saturation, lightness) adjustment zone of a single camera for detection and distance signatures. Here, first, skin correction is used in the TSL colour space, which can process information from the skin to the hand area. Second, distant features are derived from hand contours and finger points are obtained, which reduces cognitive problems in finding the peaks of one-dimensional features. The [7] article looks at the utilization of edge discovery and/or histogram equalization in pre-processing by analysing the general execution of movement acknowledgment. Nearest neighbours are used as classifiers in recognition. The system is designed to classify incoming images into one of six categories. Each class represents a different command for the machine. The gesture images were captured under controlled conditions using a webcam and a white face mask. [8] proposes a point feature extraction method based on a multilayer perceptron. Hand tan features in the YCbCr colour space are used to detect feet. By binarizing and improving the contrast of the hand image, the contour and features of the hand can be accurately extracted. [9] provides an overview of different methods for motion recognition using MATLAB. It also provides a working summary of the recognition process using edge detection and skin detection algorithms. The above pieces of literature described well contributed to Hand gesture detection but there is no literature that gives information about a service that makes interaction with the device using hand gestures only. This service enables interaction with devices through hand detection by presenting appropriate symbols to the camera. It proves particularly valuable in situations where individuals need to maintain distance and avoid physical contact due to health concerns. Notably, there are currently no existing citations or references related to contactless interaction with a device, indicating the novelty and uniqueness of this concept.

## **4. Proposed Work**

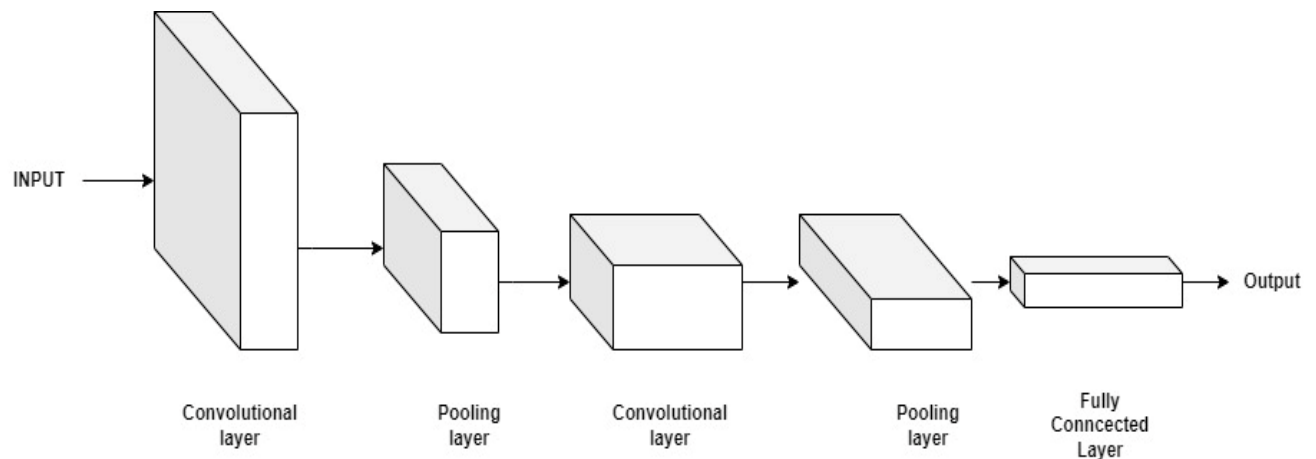
### **4.1 Contribution:**

This proposed work is mainly used in the period of a critical era where the viruses spread widely due to touching the services. This proposed work will help users be touched free where the person needs to touch anything in order to perform any action. This proposed work will read the gesture of the hand and convert it into an appropriate symbol of character or action and perform it. At first, the model is trained with all the symbols of characters and actions then with appropriate coding the model is used in an application where it can be used to enter characters and perform actions on the system.

### **4.2 Convolution Neural Network**

At the heart of a CNN lies the convolutional layer, which plays a pivotal role in the algorithm's effectiveness. Within this layer, a set of learnable filters, also known as kernels or feature detectors, is applied to the input image. As these filters convolve, they traverse the image, meticulously computing dot products between the filter weights and localized patches of the image. Through this process, a collection of feature maps is generated, adeptly capturing distinct visual patterns inherent in the input.

Referring to Figure 1, it becomes apparent that the primary objective is to capture increasingly interactive features. CNNs commonly employ multiple convolutional layers. As information traverses deeper into the network, the neurons' receptive fields expand, enabling them to encapsulate more extensive and abstract features. These intermediate feature maps are then subjected to further processing by other layer types, such as pooling layers and fully connected layers.



**Figure1. CNN Architecture**

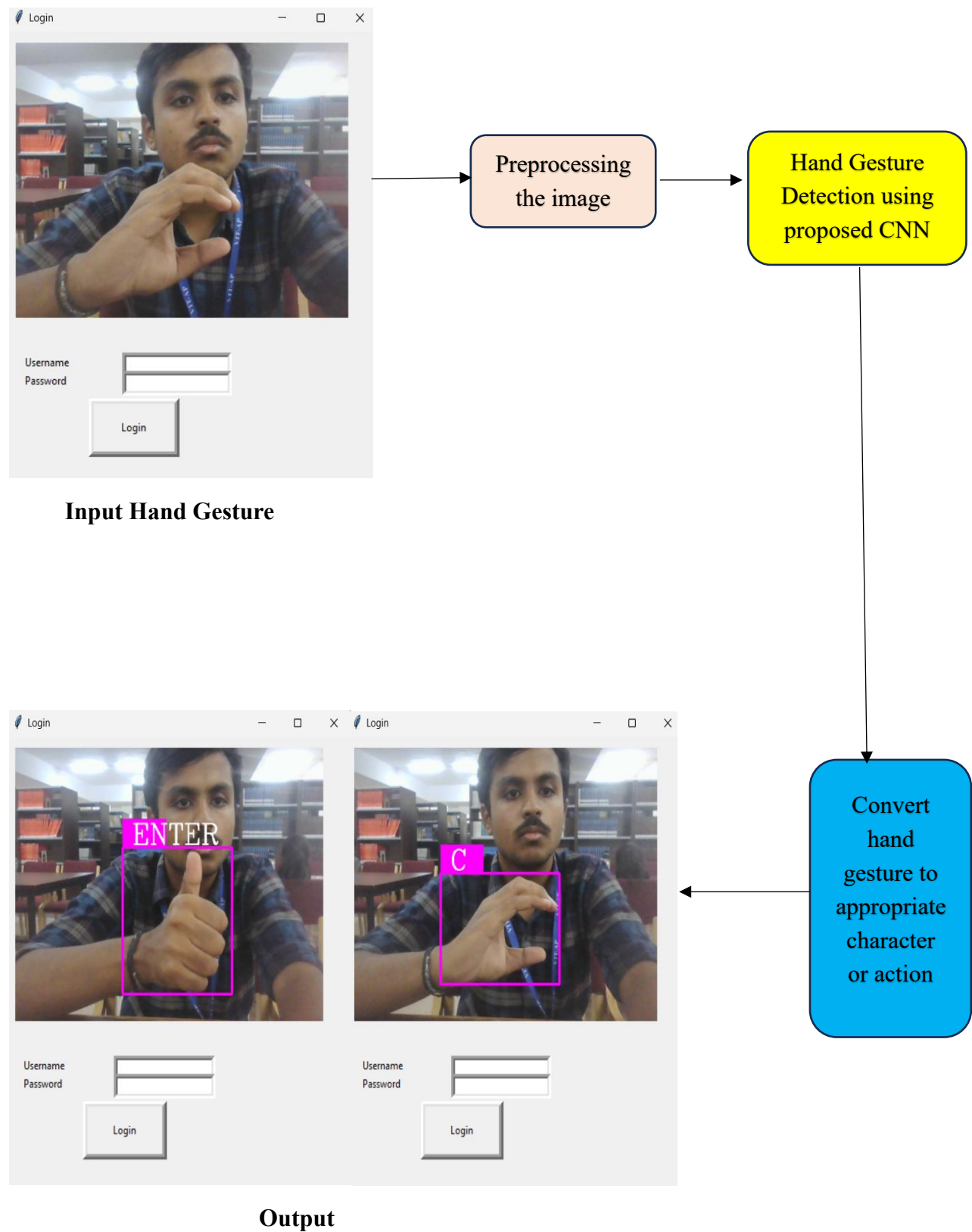
Pooling layers play a crucial role in reducing the spatial dimensions of the feature maps while retaining vital information. A prevalent technique employed within pooling layers is max pooling, which selectively preserves the maximum value within a given pooling window, effectively downsampling the feature maps.

At the conclusion of the network, fully connected layers assume responsibility for making predictions based on the extracted features. These layers establish dense connections between neurons from the previous layer and those in the subsequent layer, forming a comprehensive network. Activation functions, such as soft-max or sigmoid, are utilized within these layers to generate class probabilities or regression values, contingent upon the specific task at hand.

#### **4.3 Working flow of Proposed Architecture**

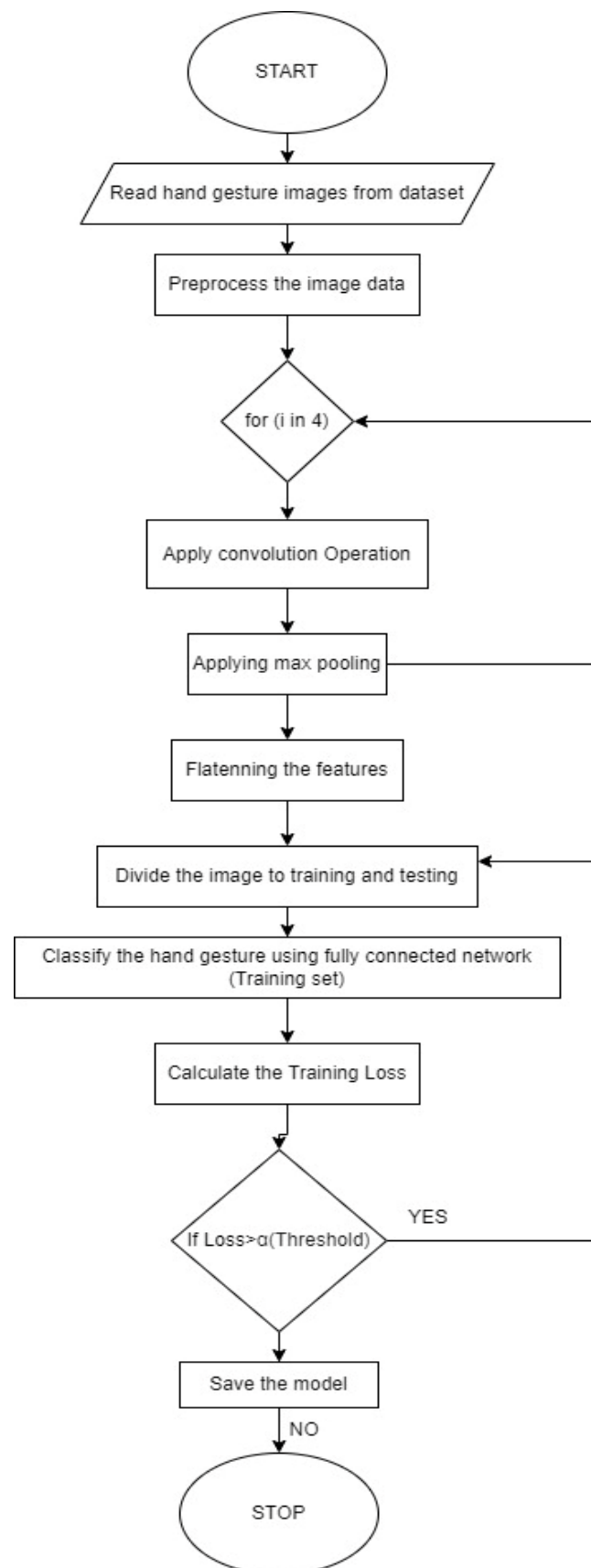
Let us now explore the architectural diagram referenced in Figure 2. At first, the hand gesture is captured. Then the image gets pre-processed. Preprocessing involves a series of steps and methods to convert raw data into a suitable format for analysis or further processing. It plays a vital role in data analysis and machine learning tasks by cleaning, organizing, and improving data quality. After preprocessing is done then the hand gesture detection is done through the CNN. Comprised of interconnected layers, CNNs automatically learn and extract essential features from the input data. The fundamental components that make up a CNN include convolutional layers, pooling layers, activation functions, fully connected layers, a loss function, and an optimizer. After detecting hand gestures using CNN now convert that

hand gesture to a particular character or action and perform it. After performing the appropriate action or showing the appropriate symbol the user can see output on the screen.



**Figure2. Proposed Architecture**

#### 4.4 Training and Testing



**Figure3. Training the model**

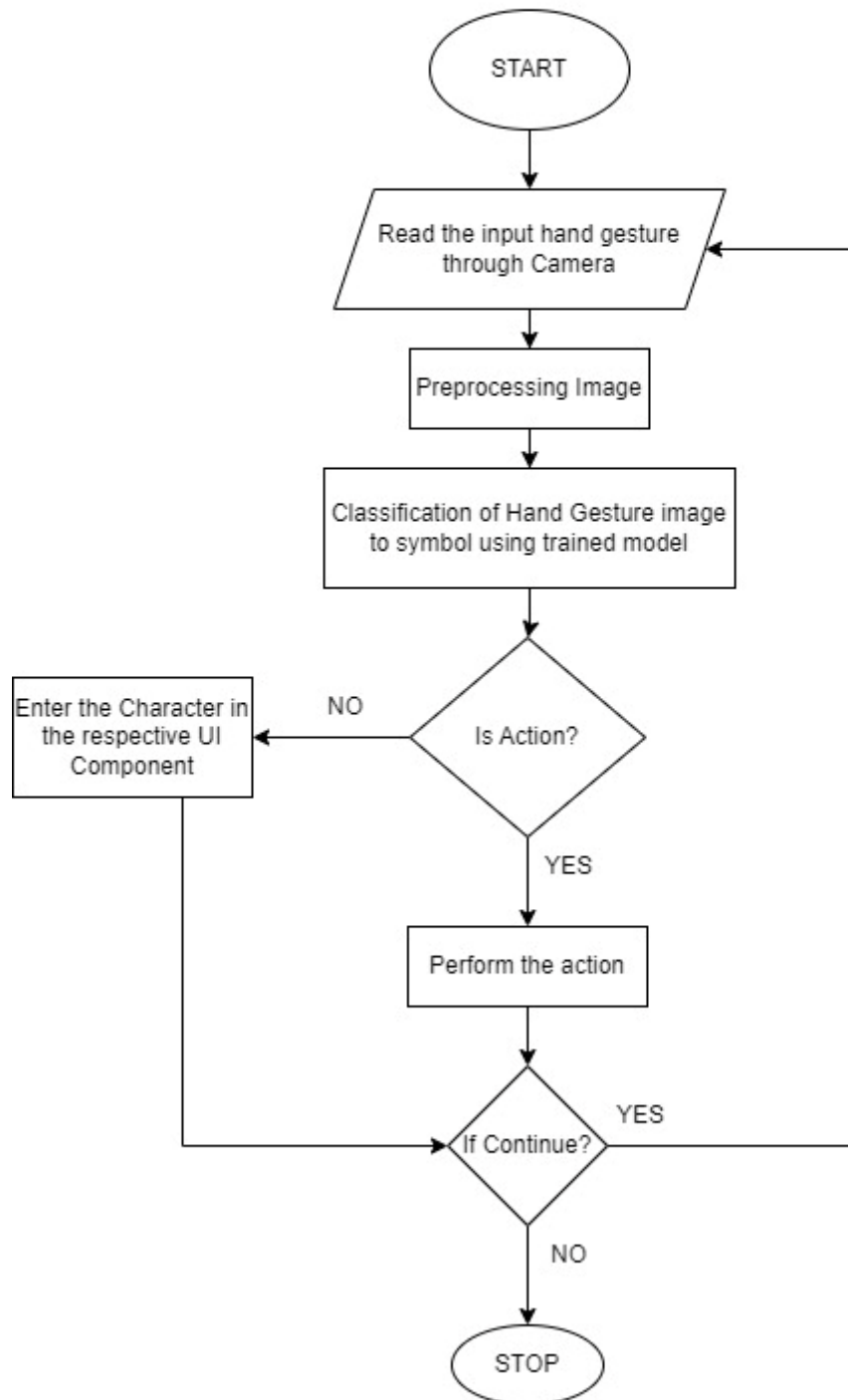
Let us now proceed to the training phase of the model, as illustrated in Figure 3. There is a set of image datasets of hand gestures defined with proper character or action. Then the device has to read the images from the given dataset. Then the images should be pre-processed in order to get the finest images. After preprocessing the data is sent to the combination of the convolution and pooling layers. Here, four combinations of convolution and pooling operation is applied. In convolution layer, the convolution operation between input image and kernel filters will be take place. Finally, it will generate the feature map in high dimensional space.

The feature map with high dimensional space is given to max pooling layer. In max pooling the dimensions of the feature map are reduced. In pooling the most prominent features are taken from the previous feature map and converted to the current feature map. Now the flattening of features is done where the 2-dimensional feature vector map is converted to a linear vector. Now divide the image into training and testing parts. The fully connected network allows for efficient pattern recognition and classification tasks. By connecting all neurons in a layer to all neurons in the subsequent layer, the network can learn complex relationships and extract relevant features from the input data, ultimately leading to accurate classification results.

This type of network architecture is commonly used in various machine learning and deep learning applications, including image recognition, natural language processing, and more. Now calculation of training loss takes place where the difference between the generated outcome and the anticipated outcome is taken. Conversely, if the loss is below or equal to the specified threshold ( $\alpha$ ), indicating that the model has achieved satisfactory performance, the system saves the trained model to retain its current state. After saving the model, the flow stops, and the system is ready to be deployed for practical use. By periodically evaluating the loss and updating the model during the loop, the system becomes more accurate and effective in its gesture recognition and interaction capabilities, leading to an enhanced user experience.

We shall now transition to the testing phase, which involves capturing hand gestures and mapping them to corresponding actions, as indicated in Figure 4. Let's start the process by taking the hand gesture input through the device camera. After capturing the hand gesture image then preprocess the image where the image data is cleaned properly and gets best out of worst. After preprocessing is done then the refined image of the hand gesture is classified into appropriate symbols using the trained model. If the symbol is an action then perform the appropriate action like enter, backspace, next, and submit, where enter option enters the shown character into the text box, the backspace option backspaces a character once, the next option tabs from the current text box to the upcoming text box, and submit option submits the user input. After performing action or showing character if the user wants to continue the operations then the loop will go to image hand gesture input otherwise it stops the execution. So this is the testing part of the proposed work.

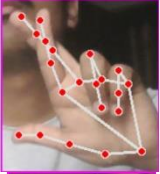


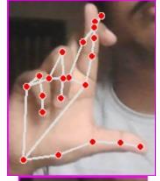



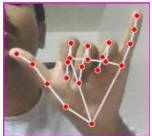

## Capturing Hand Gesture and Mapping to action





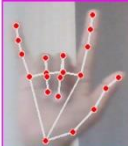

**Figure4. Testing the model**



SYMBOL	CHAR	SYMBOL	CHAR	SYMBOL	CHAR
	0		9		I
	1		A		J
	2		B		K
	3		C		L
	4		D		M
	5		E		N
	6		F		O
	7		G		P
	8		H		Q

SYMBOL	CHAR	SYMBOL	CHAR		CHAR
	R		X		U
	S		Y		V
	T		Z		W

**Table1. Symbols for each Character**

SYMBOL	ACTION
	ENTER
	BACKSPACE
	NEXT
	SUMBIT

**Table2. Symbols for each Action**

## **5. RESULT AND ANALYSIS**

### **5.1 Dataset Details**

<b>Used for Dataset</b>	<b>Training</b>	<b>Testing</b>
<b>Alphabet Dataset</b>	12988	1000
<b>Numeric Dataset</b>	5177	1000
<b>Actions Dataset</b>	2000	1000

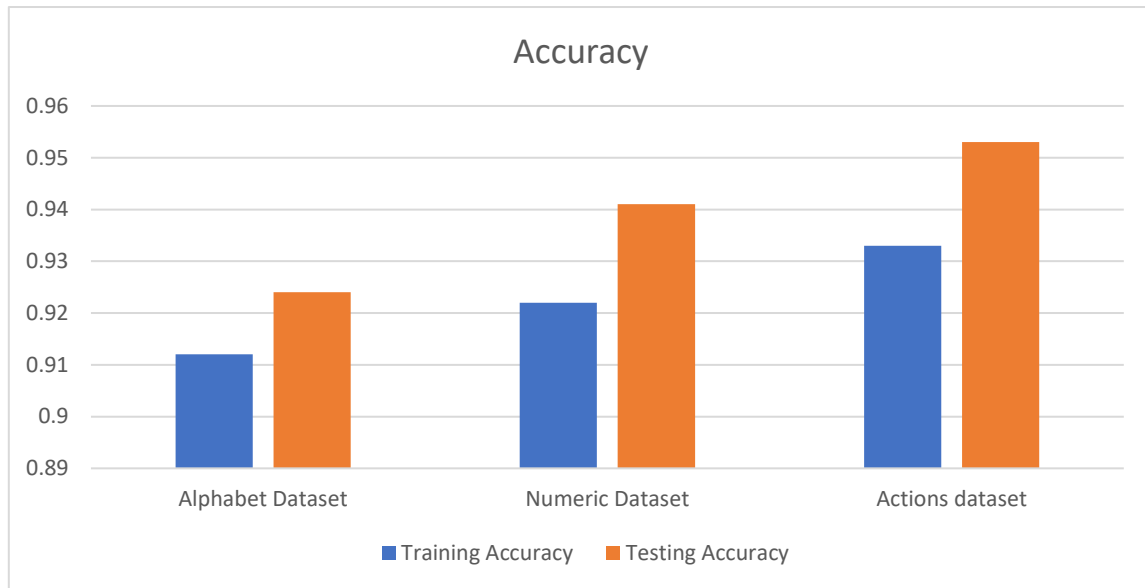
**Table3. Dataset Details**

Table 1 and Table 2 present a comprehensive symbol table, which includes 26 distinct symbols allocated for representing each of the 26 alphabets in the English language. Moreover, the symbol table extends its utility to encompass numeric data as well, with 10 unique symbols assigned to represent the numbers 0 through 9. It also defines four hand gestures for specific actions, enabling touchless interaction with devices and applications. For training 12988 images are used from alphabet dataset, 5177 images from numeric dataset and 2000 images from actions dataset making a total count of 20165 images for training. For testing 1000 images each are used from alphabet dataset, numeric dataset and actions dataset making a total count of 3000 images for testing.

### **5.2 Results and Discussions**

<b>Accuracy Dataset</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
<b>Alphabet Dataset</b>	0.912	0.924
<b>Numeric Dataset</b>	0.922	0.941
<b>Actions Dataset</b>	0.933	0.953

**Table4. Accuracy Details**

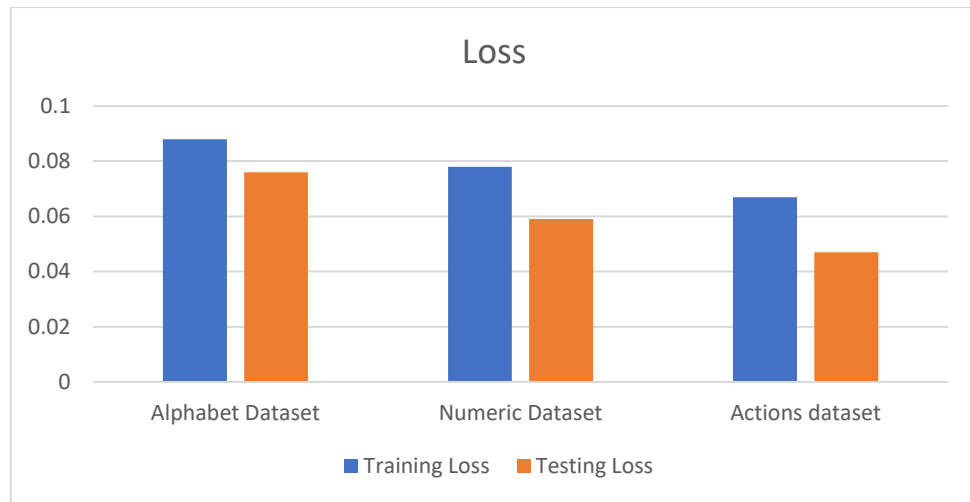


**Figure5. Accuracy Details Graphical Representation**

The Table4 and Figure5 displays the accuracy results for three different datasets. The datasets are labelled as "Alphabet Dataset," "Numeric Dataset," and "Actions Dataset." For each dataset, two types of accuracy metrics have been recorded: "Training Accuracy" and "Testing Accuracy". Alphabet Dataset have a training accuracy of 0.912 and testing accuracy: 0.924. Numeric Dataset have a training accuracy of 0.922 testing accuracy of 0.941 actions dataset have a training Accuracy of 0.933 and testing accuracy of 0.953. By examining the tabulated data, it becomes evident that the "Actions Dataset" attains the most elevated training and testing accuracy figures, with the "Numeric Dataset" ranking second and the "Alphabet Dataset" coming in third. This implies that the model exhibits superior predictive capabilities for the "Actions Dataset" when compared to the other two datasets.

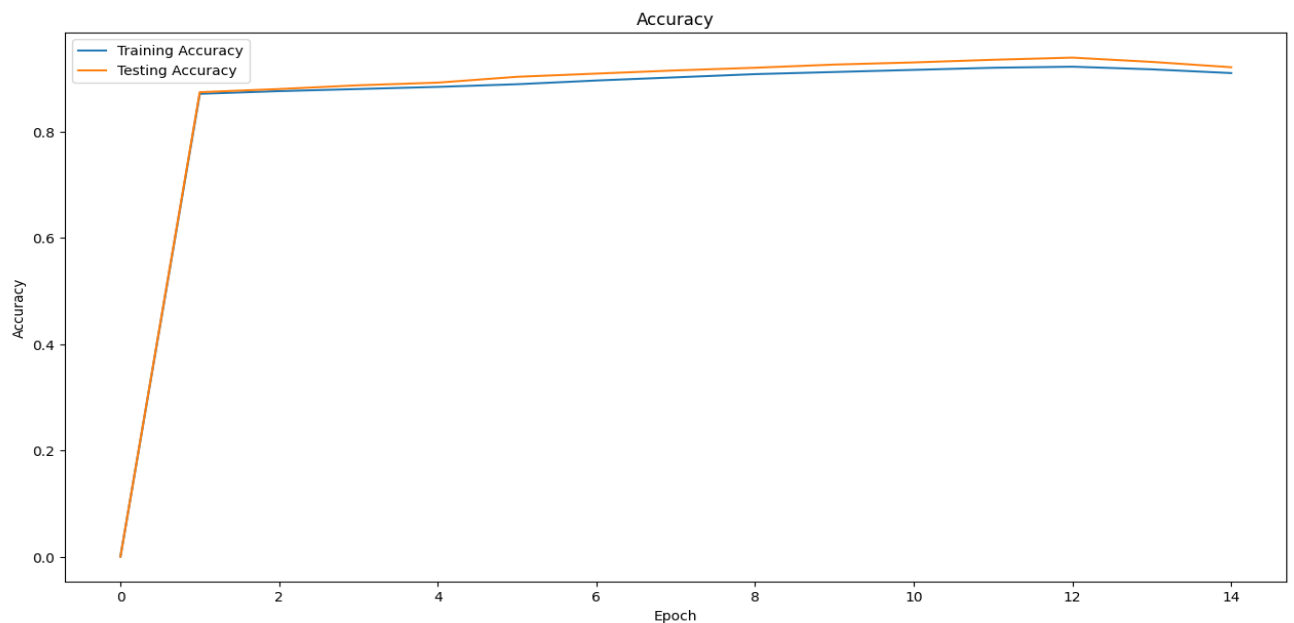
Dataset \ Loss	Loss	
	Training Loss	Testing Loss
Alphabet Dataset	0.088	0.076
Numeric Dataset	0.078	0.059
Actions Dataset	0.067	0.047

**Table5. Loss Details**



**Figure6. Loss Details Graphical Representation**

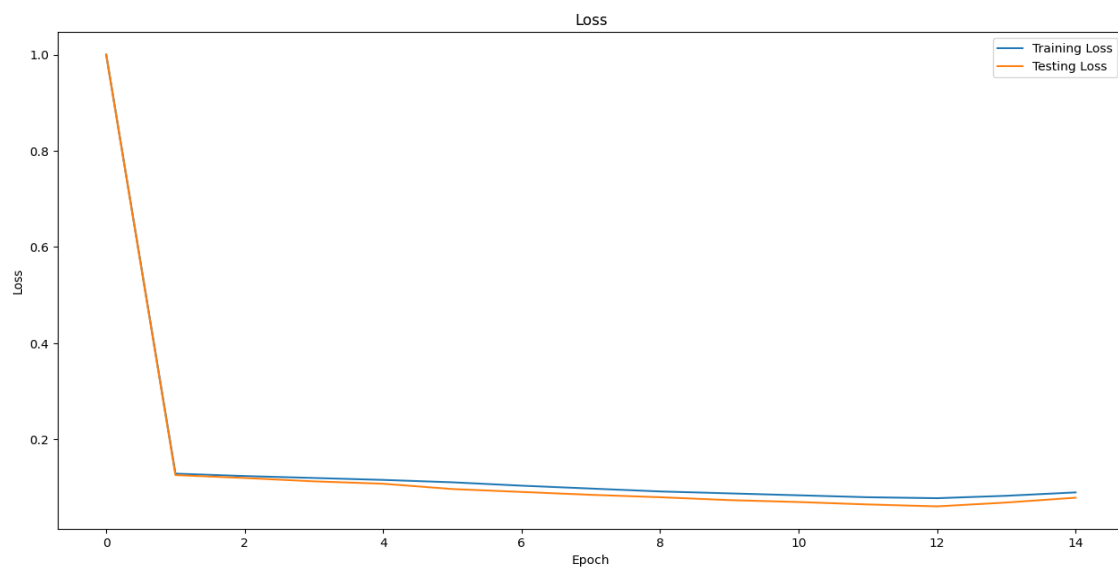
The Table5 and Figure6 provides a summary of the loss results for three distinct datasets: the "Alphabet Dataset," the "Numeric Dataset," and the "Actions Dataset." Each dataset is evaluated using two essential metrics: "Training Loss" and "Testing Loss". Alphabet Dataset have training loss of 0.088 and testing loss of 0.076. Numeric dataset have training loss of 0.078 testing loss of 0.059. Actions Dataset have a training loss of 0.067 and testing loss of 0.047. From the table, it can be observed that the "Actions Dataset" has the lowest training and testing loss values, followed by the "Numeric Dataset," and then the "Alphabet Dataset." This suggests that the model performs better in predicting outcomes for the "Actions Dataset" compared to the other two datasets. Evaluating and optimizing loss values is an essential step in fine-tuning machine learning models to achieve higher accuracy and better generalization to new data.



**Figure7. Accuracy**

Figure 7 depicts the graph of the model's training and testing accuracy at different epochs, with the x-axis denoting the epoch numbers from 0 to 14 and the y-axis representing the

corresponding accuracy values. The blue line represents the model's training accuracy, which starts at 0 and gradually increases with each epoch. This indicates that the model is effectively learning from the training data, leading to improved accuracy over time. The orange line represents the model's testing accuracy, which also begins at 0 and shows a similar upward trend with increasing epochs. This suggests that the model's performance is not only improving on the training data but also on unseen data, demonstrating its ability to generalize well. Throughout the graph, both accuracy lines show consistent and steady progress, indicating a positive learning curve. This implies that the model is progressively fine-tuning its predictions, leading to enhanced accuracy on both the training and testing datasets.



**Figure8. Loss**

Figure 8 graph depicts the training and testing loss values of a model across different epochs. The x-axis represents the epochs, ranging from 0 to 14, while the y-axis shows the corresponding loss values. The blue line illustrates the model's training loss at each epoch. As the epochs progress, the training loss consistently decreases, indicating that the model is effectively learning from the training data. The downward trajectory of the line suggests that the model is continuously refining its predictions and minimizing errors during the training process. The orange line signifies the model's testing loss at each epoch. Like the training loss, the testing loss also demonstrates a downward trend. This suggests that the model generalizes well to new, unseen data, as indicated by the reduction in loss values during the testing phase. The ability to reduce testing loss reflects the model's capability to make accurate predictions on data it has not encountered during training. In conclusion, the line graph provides valuable insights into the model's learning progress, showcasing the reduction in loss values throughout the training and testing phases. The graph's trends reveal the model's ability to learn from the data, adapt its predictions, and achieve higher accuracy on both training and testing datasets, making it an essential visualization for assessing the model's overall performance.

## **6.REFERENCES**

- 1) A. Gupta, Y. Kumar and S. Malhotra, "Banking security system using hand gesture recognition," 2015 International Conference on Recent Developments in Control, Automation and Power Engineering (RDCAPE), Noida, India, 2015, pp. 243-246, doi: 10.1109/RDCAPE.2015.7281403.
- 2) P. Surekha, N. Vitta, P. Duggirala and V. S. S. Ambadipudi, "Hand Gesture Recognition and voice, text conversion using," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp. 167-171, doi: 10.1109/ICAIS53314.2022.9743064.
- 3) D. G. León et al., "Video Hand Gestures Recognition Using Depth Camera and Lightweight CNN," in IEEE Sensors Journal, vol. 22, no. 14, pp. 14610-14619, 15 July 2022, doi: 10.1109/JSEN.2022.3181518.
- 4) H. D. Alon, M. A. D. Ligayo, M. P. Melegrito, C. Franco Cunanan and E. E. Uy II, "Deep-Hand: A Deep Inference Vision Approach of Recognizing a Hand Sign Language using American Alphabet," 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 2021, pp. 373-377, doi: 10.1109/ICCIKE51210.2021.9410803.
- 5) X. Dong et al., "A Static Hand Gesture Recognition Model based on the Improved Centroid Watershed Algorithm and a Dual-Channel CNN," 2018 24th International Conference on Automation and Computing (ICAC), Newcastle Upon Tyne, UK, 2018, pp. 1-6, doi: 10.23919/IconAC.2018.8749063.
- 6) L. Gu, X. Yuan and T. Ikenaga, "Hand gesture interface based on improved adaptive hand area detection and contour signature," 2012 International Symposium on Intelligent Signal Processing and Communications Systems, Tamsui, Taiwan, 2012, pp. 463-468, doi: 10.1109/ISPACS.2012.6473534.
- 7) R. Lionnie, I. K. Timotius and I. Setyawan, "An analysis of edge detection as a feature extractor in a hand gesture recognition system based on nearest neighbor," Proceedings of the 2011 International Conference on Electrical Engineering and Informatics, Bandung, Indonesia, 2011, pp. 1-4, doi: 10.1109/ICEEI.2011.6021611.
- 8) C. Yu, X. Wang, H. Huang, J. Shen and K. Wu, "Vision-Based Hand Gesture Recognition Using Combinational Features," 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Darmstadt, Germany, 2010, pp. 543-546, doi: 10.1109/IHMSP.2010.138.
- 9) S. K. Yewale and P. K. Bharne, "Hand gesture recognition using different algorithms based on artificial neural network," 2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC), Udaipur, India, 2011, pp. 287-292, doi: 10.1109/ETNCC.2011.6255906.

- 10) J. Kunjumon and R. K. Megalingam, "Hand Gesture Recognition System For Translating Indian Sign Language Into Text And Speech," 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2019, pp. 14-18, doi: 10.1109/ICSSIT46314.2019.8987762.
- 11) Ali, Ahmed & Elmisery, Fathy & Mostafa, Ramadan & Hussein, Mohammed. (2014). Motion Control of Robot by using Kinect Sensor. Research Journal of Applied Sciences, Engineering and Technology. 8. 1384-1388. doi: 10.19026/rjaset.8.1111.