Machine Learning and Mediapipe Assisted Sign Gesture-based Smart Keyboard for Deaf and Blind People

Arsalan Ali
Department of Biomedical Engineering University of Engineering and Technology,
Lahore, 54890, Pakistan
2019bme111@uet.edu.pk

Muhammad Hamza Zulfiqar
Department of Biomedical Engineering University of Engineering and Technology,
Lahore, 54890, Pakistan
hamzazulfiqar@uet.edu.pk

Muhammad Qasim Mehmood
Micro Nano Lab, Electrical Engineering Department, Information Technology University (ITU) of the Punjab, Ferozepur Road, Lahore 54600, Pakistan
qasim.mehmood@itu.edu.pk

2024 5th International Conference on Advancements in Computational Sciences (ICACS) | 979-8-3503-9478-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICACS60934.2024.10473252

***Abstract*— There are 1.1 billion deaf and 2.2 billion blind people, and many people have speech impairments. These disabilities cause them to be unable to communicate with others and use modern technologies. This paper presents a real-time smart touchless keypad by which people with blindness, deafness, and speech impairments can interact with the computer with the help of 37 hand gestures using a single camera. The presented solution uses a hand tracking module from the mediapipe library for feature extraction, a random forest algorithm for the classification of gestures, and a PyAutoGUI library for controlling the mouse and keyboard according to the classified hand gestures. The presented solution has achieved an accuracy of 99.66% on the test set, indicating that this system can also detect gestures for communication with deaf or speech-impaired people. Overall, this system employs computer vision and machine learning techniques to improve the lives of the deaf, blind, and mute people.**

Keywords—sign gesture, mediapipe, smart keyboard, deaf, blind, human-computer interaction

I. INTRODUCTION

Communication is an effective tool to share ideas, thoughts, and feelings. Communication between humans makes life easier. But there is a gap because, unfortunately, there is a community of deaf and visually impaired people who cannot communicate with other people. According to the World Health Organization, 1.1 billion people are living with deafness, and 2.2 billion people are living with visual impairments [1], [2]. Many deaf people even cannot speak as well. Advancements in technology have made our lives easier. Still, technology has not proved so beneficial for deaf, mute, and visually impaired people because they are unable to communicate with other people and even cannot interact with modern technologies like computers [3]. It is necessary to communicate with modern technology because technology is an integral part of society nowadays. Technology assists society in every area of life like. Google Maps helps millions of people in navigation. In the same way, a keypad is used to write messages for the purpose of communication. But to interact with technology, there is a need for communication between humans and machines like computers. So, keeping this need for communication for deaf and visually impaired people, many sign languages have been introduced in the world by which deaf people can communicate with the other people and also with technology. Almost all the sign languages are in the form of gestures or visuals. These are different according to sociolinguistic demography [4].This solution needs an interpreter who understands the signs of that language. This is the costly solution for having interpreter and also makes deaf and visually impaired people dependent on others for communication. So there is need of technological solution which is cost effective and makes the effected people independent for interpersonal communication and communication with modern technology.

Many solutions have been proposed, but there are three famous ones. The first solution is based on computer vision, the second is a sensor-based solution, and the third is voice assistant-based technology that only works for blind people, not for deaf people. The first solution processes images or videos to determine the relevant sign. Mobile cameras or other web-common cams can be used to take images. On the other hand, in the second solution, sensors are placed on the hand or fingers, and then the data from the sensors is analyzed to determine the relevant sign of the hand [5]. As the second solution is a sensor-based approach, also known as the data-glove approach, in which we mostly detect the hand position, angle of fingers bending, wrist movement etc. [6]. The third solution is a voice assistant-based solution which helps blind people to communicate with technology like Apple Siri.

J. Rekha used a multiclass nonlinear support vector machine to classify the signs of Indian Sign Language with a 91.3% rate of recognition [7]. R. Sutjiadi presented the deep convolution neural network that can classify the 26 signs of Indonesian Sign Language with an accuracy of 75.38%[8]. Jyotishman Bora presented a framework that can classify only 9 static gestures of Assamese Sign Language using feed-forward neural network with an accuracy of 99%[9]. Yutong Gu presented a classification model that can classify the 26 dynamic signs of American Sign Language with an average accuracy of 74.8% [10]. A. M. Buttar used a hybrid approach with Mediapipe Holistic model and YoloV6 to classify signs with an accuracy of 92% and 96%, respectively [11]. Francisco Morillas-Espejo presented a solution named sign4all based on a deep convolution neural network in which they recognized the sign using the ResNet50 framework and got an accuracy of 79.96% on RGB images. They also designed virtual avatars to write letter by letter [12].

M. Rinalduzzi presented a magnetic positioning system-based solution where the wearable transmitting nodes are placed in a specific 3D space, and the receiving nodes, after receiving the message from the transmitting nodes, classify the 24 signs of Sign Language with an accuracy of 97% [13]. K. Bhat presented a smart glove employing flex sensors to detect finger movement. The smart glove interprets the hand's signs using flex sensors and converts them into meaningful messages that can be heard with the Android app

[14]. Astha Dogra designed a glove with IMU sensors on the finger to track the movement of fingers and interpret the sign. After detecting the sign , it is displayed on LCD via Bluetooth[15]. This is costly because it requires sensors and placements on the wearables. Additionally, most present sensors are made of plastic, which also cause the e-pollution.

One of the solutions that is designed to help blind people is a voice assistant-based solution. This solution does not work for deaf and mute people. Many general solutions like SIRI, Alexa, Google Allo, Microsoft Cortana, and so on can be employed to help visually impaired people [17]. A. KARTHIK presented a solution that read environmental messages using Raspberry Pi and OCR sensor and gives response in speech [18]. Regardless of the growing trend for using voice assistants, researchers raise many issues. One is "how to use voice assistant and comprehend its response?" by blind people because every blind person cannot use voice assistant[19]. Privacy of personal user data is a rising concern regarding the voice assistant ecosystem because research shows that in the voice assistant ecosystem, users' personal data is stored temporarily or permanently for commercial or other purposes [20]. Voice assistants are unable to recognize voice commands accurately in noisy environments, which makes the solution ineffective [21].

We propose a real-time machine learning-based multipurpose solution in which we have devised an intelligent touchless keypad in which people with deafness and visual and speech impairments can interact with computers or devices. This solution can be used to recognize the signs, which can help interpersonal communication and remove the need for an interpreter. In this solution, we have introduced new characters which can be used for specific functions and indications. We have achieved a high accuracy of 99.66% on the test set, so our solution is consistent and reliable.

In the first section of paper, we have discussed the need of smart keyboard for blind and speech impaired people in introduction section. In second section, we have discussed about data collection, data preprocessing, model training,hyperparameters tuning and mapping the labels with keyboard keys and mouse cursor using PyAutoGUI.In the third section we have validated the model with the help of results obtained using K-fold cross-validation,learning curves,confusion matrix,F1 score,precison,recall and the comparative analysis of results with related solutions.In fouth and fifth section we have discussed the conclusion and future works respectively.

II. METHODOLOGY

In this section, we have presented the solution implementation and how it will help the blind and speech impaired people to interact with technology and its practical implementation using PyAutoGUI. First of all ,the images are collected and processed with the mediapipe. After which, we trained the model and then took the predictions. Fig. 1 shows the overall methodology of the solution from data collection to the training of the model and the predictions from the model.

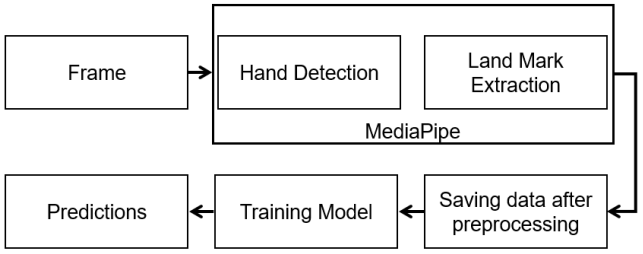

Fig. 1. Flow diagram of overall working from data collection to making predictions.

A. Data Collection

Firstly, the 37 gestures are selected, as shown in Fig. 2, from different sign languages like American Sign Language, Pakistani Sign Language, etc. Single-hand gestures are selected to make the computations fast. These signs are reserved for specific keypad keys like 0 to 25, which are mapped with A, B,…,Z in the same way the remaining signs are mapped with other useful keys. 7,400 images were collected from 200 photos for each gesture to avoid the data imbalance problem.

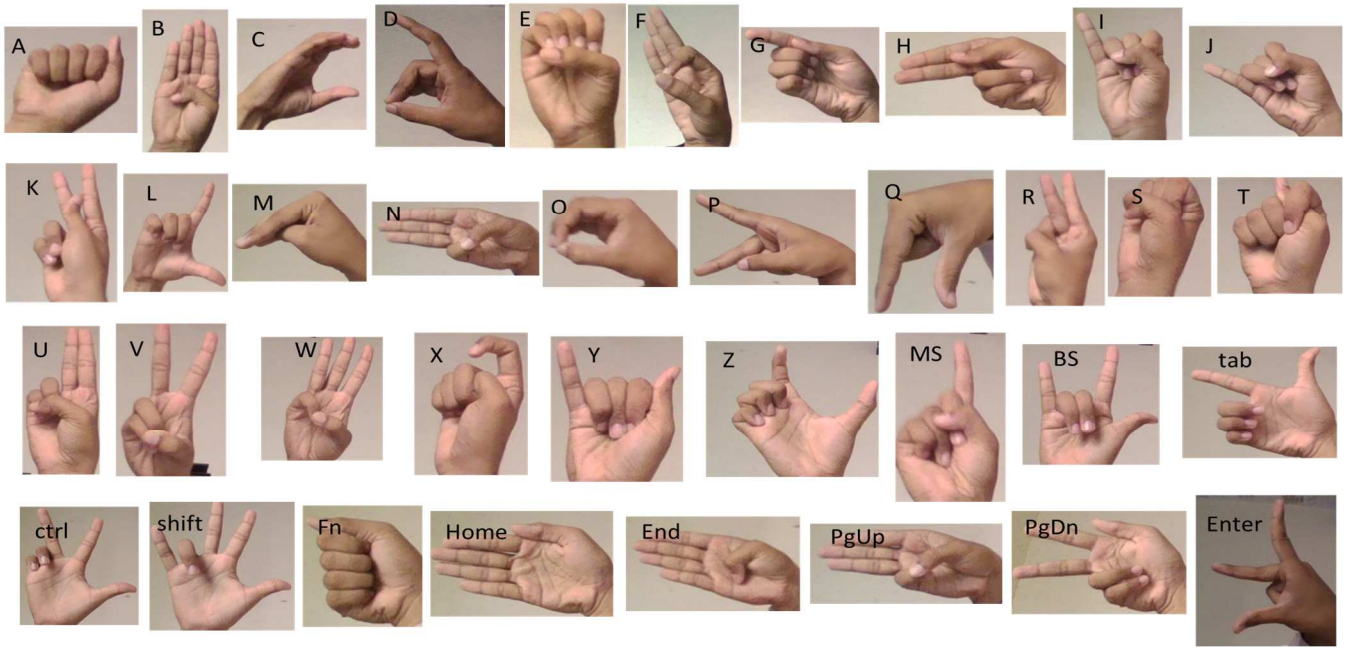

Fig. 2. All 37 signs were used for classification.

B. Dataset Creation

We have processed the image using the hand landmark model bundle from the mediapipe framework developed by Google. As shown in Fig. 3, the hand landmark model bundle detects the 21 unique landmarks on the hand and gives their coordinate values.

We have passed the images to mediapipe by the OpenCV library, which is usually used for computer vision tasks. OpenCV represents the images in BGR format, but machine learning and learning expect the images to be in RGB format. So, with the help of OpenCV, we have converted the images from BGR to RGB format. Then RGB images are fed into the hand model bundle provided by the mediapipe.

This framework gives the three coordinates of each landmark: x, y, and z. But we have only used x and y coordinates. As there are 21 landmarks in the hand and have have selected the x and y coordinate. So there will be 44 features or keypoints which are calculated using (1). We have saved the features's data from landmarks of every image, kept the data in empty arrays, and made a single list of arrays. Then, using the labels, we defined the dictionary and exported the data using the pickle library provided by Python. As all our signs contain a single hand (R), the number of keypoints in one image are:

$$Total\ keypoints\ in\ one\ image = No\ of\ hand \times Keypoints\ in\ hand \times dimensions \quad (1)$$

$$Total\ keypoints\ in\ one\ image = 1 \times 21 \times 2 = 44 \quad (2)$$

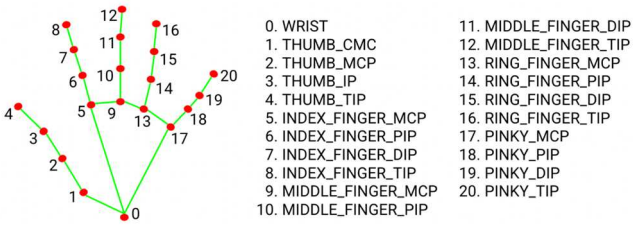

Fig. 3. List of hand landmarks detected by Hand Land Mark Model from mediapipe by Google.

C. Data Preprocessing

As the data is collected from raw images, there is a chance of null values and noise. So, to avoid bias in the model, data preprocessing is needed. The data is preprocessed, excluded, or replaced with the missing features with the mean value—also, the scaled so that data becomes consistent. All data preprocessing is done using the Pandas library, which is also open-source.

D. Sampling

The dataset is split into two sets: the training and test sets. The training set consists of 80% data, and the test set contains 20%. We ensure that there must be an equal amount of classes in the test set to avoid data imbalance problems.

E. Modeling

We have trained all famous machine learning algorithms used for classification on the training set and have selected the random state 42 so that the results of all the classifiers remain consistent. We have preferred machine learning over deep learning frameworks because they are computationally expensive, like deep CNN, which are computationally expensive and takes a lot of time for training on normal GPU. We have achieved the desired accuracy on machine learning algorithms, which are also easy to train. The algorithms used are random forest, support vector machine, decision tree, the Gaussian Naive Bayes classifier, etc., after which we have performed k-fold cross-validation and learning curves to check the overfitting. Then, we used the confusion metrics to analyze each class's classification and also calculated each class's F1 score, recall, and precision.

F. Hyperparameter Tuning

We have adjusted the hyperparameters of the classifiers, like for the random forest classifier, the n-estimator=100, random_state: 42, verbose=0, etc., to enhance accuracy.

G. Mapping Predicted Labels with Keyboard Keys and Cursor

After getting the classification model's prediction, we activated the keys mapped with that class using PyAutoGUI, which Al Sweigart developed. PyAutoGUI is a Python library that is used to control mouse and keyboard keys.

III. Results and Discussion

Table I. shows the accuracy of the classifiers that performed well on test sets after training on the training set:

TABLE I. Accuracy of different classification models

Classifier	Accuracy on test set
Random Forest	99.66%
Support Vector Machine	98.38%
Decision Tree	99.12%
Gaussian Naive Bayes	93.18%

For further process, we have chosen the random forest algorithm not only due to its performance on the test set but also due to its robustness and ability to handle complex and noisy data. As the camera quality of every webcam or mobile is not the same, there is a chance of noisy data, so we have further persuaded the random forest classifier. The random forest is also not so sensitive to hyperparameters. But, hyperparameter tuning is required for optimal results.

The hyperparameters of the trained random forest classifier are:

bootstrap: True
ccp_alpha: 0.0
class_weight: None
criterion: gini
max_depth: None
max_features: sqrt
max_leaf_nodes: None
max_samples: None
min_impurity_decrease: 0.0
min_samples_leaf: 1
min_samples_split: 2
min_weight_fraction_leaf: 0.0
n_estimators: 100
n_jobs: None
oob_score: False

random_state: 42
verbose: 0
warm_start: False

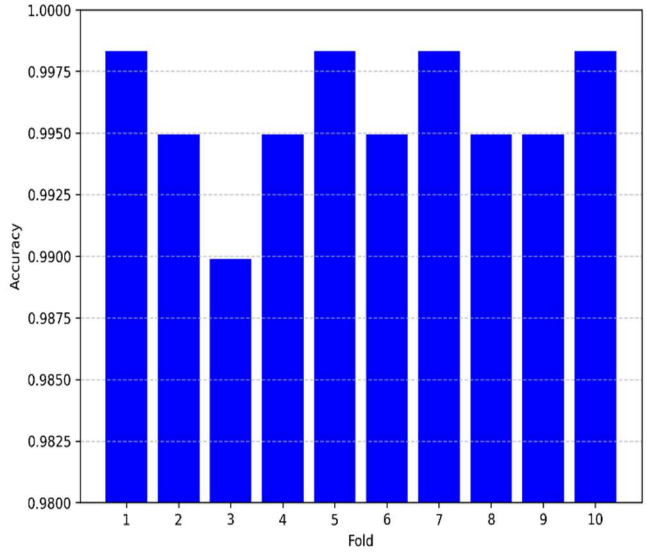

Fig. 4. K-fold cross-validation results when k=10.

A. K-fold Cross-validation

The training data is divided into 10 folds, and each fold acts as a validation set to evaluate the random forest classifier. The results show that the model has performed consistently and pretty accurately on all folds shown in Fig. 4.

B. Learning Curves

Learning curves are the function of model performance on training and validation set to training data size. As shown in Fig. 5, the training and validation accuracy are close to one. This suggests that the model has not only performed well on the training and validation sets but is also very well fitted to training data and generalized to unseen cases.

C. Confusion Matrix

The confusion matrix is used to check how all the classes are classified. Fig. 6 shows that the model performs well on the testing set, and almost all the classes are on the main diagonal and classified correctly.

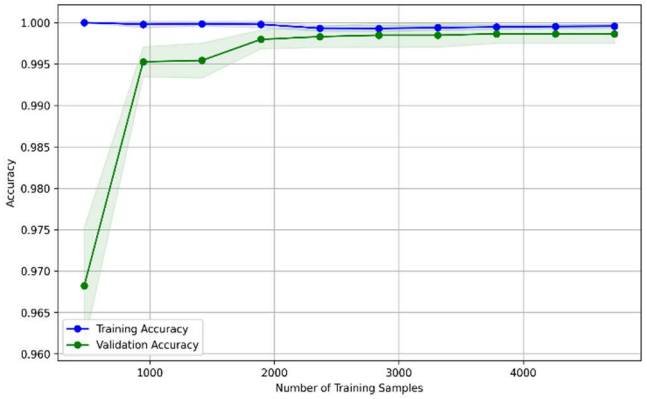

Fig. 5. The learning curve shows the training accuracy and validation accuracy.

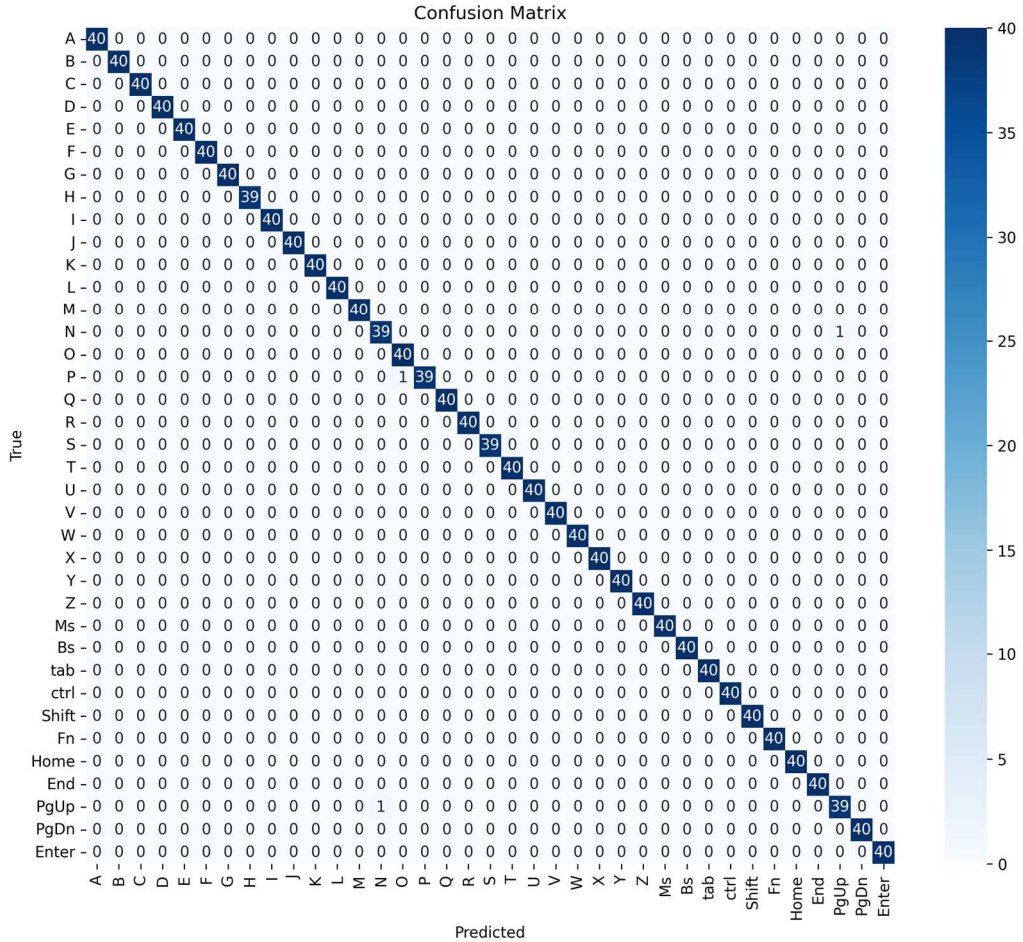

Fig. 6. Confusion matrix of all 37 gestures/signs.

D. F1 Score,Precision and Recall:

The F1 score is another precision metric, and its higher value shows that the model performed well. It is calculated using (3):

$$F1\ score = \frac{2 \times (Precision \times Recall)}{Presion + Recall} \quad (3)$$

Equation 3 is obtained using (4) and (5) discussed below. Fig. 7 shows that the F1 of every label from 0 to 36 is nearly equal to 1, which means the model has performed well on the testing set.

Precision in (3) is another interesting metric to see the accuracy of positive predictions. Its equation is:

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

Where TP and FP are the number of true positive(output correctly predicted as positive) and false positive(output is incorrectly predicted as positive where they are negative), respectively.

Recall in (3) is the metric which shows the sensitivity or true positive rate. Its Equation is:

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

Where FN is the number of false negatives (the number of instances incorrectly predicted as negative but actually positive), both precision and accuracy are extracted from the confusion matrix.

In Fig 7, recall and precision are almost equal to 1, and average accuracy is 0.9966, equivalent to 99.66%.

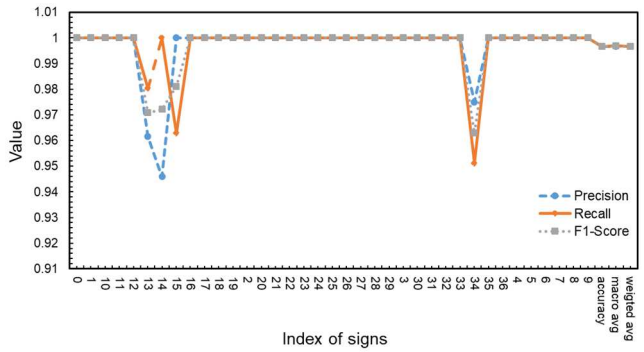

Fig. 7. The graph shows the precision, recall, and F1 score of all 37 signs.

E. Accessing Keypad with Camera

A web camera shows gestures; our system takes the input frame, and our model classifies which sign it is. Then, PyAutoGUI will press that key on the keyboard. In Fig. 8 (a), as we want to write 'A', we have shown the 'A' sign to the camera, and as expected, the prediction is the letter 'A', so the PyAutiGUI has clicked the 'A,' and we have also got alphabet 'A' in our text document. We have added a delay of 1 second between predictions of alphabets only so that there should be a reasonable control for the user to write. In Fig. 8 (b), as we want to write 'C' so, we have shown the 'C' sign to the camera, and as expected, the prediction is the letter 'C', so the PyAutiGUI has clicked the 'C' and we have also got alphabet 'C' in our text document. In Fig. 8 (c), as we want to control the cursor, we have shown the mouse sign, and as expected, the prediction is also mouse, so PyAutoGUI gives the control hand via camera; resultantly, we are controlling the cursor with the sign.

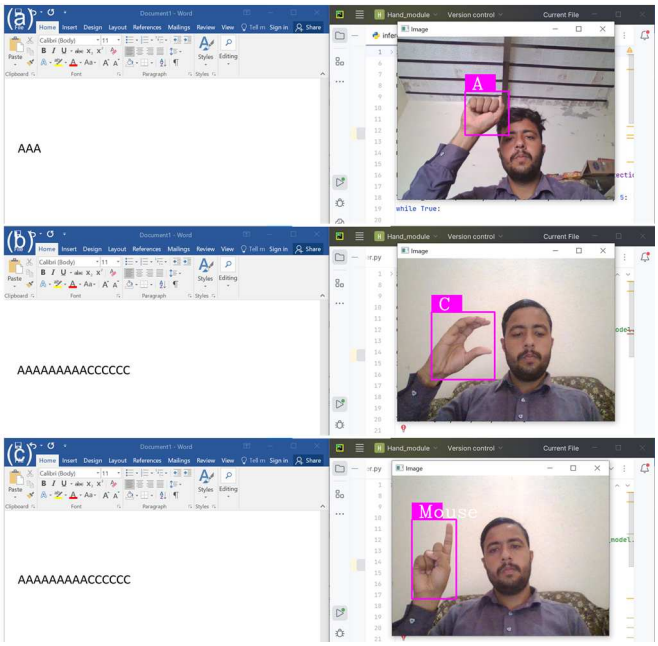

Fig. 8. Keypad and cursor control. (a) Writing A with the help of a sign. (b) Writing C with the help of sign. (c) Controlling the cursor with the use of a sign.

F. Comparative Analysis:

Table II shows that most solutions have 26 signs, whereas our solution has 37 signs. Comparative analysis shows that we have achieved a decent accuracy of 99.66%, even with 37 signs, distinguishing us from the present solutions.

TABLE II. COMPARATIVE ANALYSIS OF DIFFERENT PRESENT SOLUTIONS WITH OUR SOLUTION.

Sign Langauge	Author [Ref]	Number of Signs	Approach	Accuracy	Year
Indonesian Sign Language	R. Sutjiadi [8]	26	Convolution Neural Network	75.38%	2023
Indian Sign Language	J. Rekha [7]	26	Non-linear Support Vector machine	91.3%	2011
American Sign Language	Yutong Gu [10]	26	CNN with fully connected layer	74.8%	2022
Assamese Sign Language	Jyotishman Bora [9]	9	Feed forward neural network	99%	2022
Spanish Sign language	Francisco Morillas-Espejo [12]	26	ResNet50	79.96%	2023
American Sign Language	A. M. Buttar [11]	26	LSTM with Mediapipe	92%	2023
		26	LSTM with YoloV6	96%	2023
Signs from different Sign Langauages	Our solution	37	Random forest Classifier with mediapipe	99.66%	2023

IV. CONCLUSION

The presented solution uses a random forest algorithm trained with a vast range of signs, which are 37. These signs are also mapped with the keyboard so anybody can control the computer with our touchless solution. We have achieved an accuracy of 99.66%, and our model is consistent and reliable. The trained model is lightweight and computationally less expensive, making it suitable for Android apps or any other platform deployment. So, by analyzing the images using

MediaPipe, there is no need to train state-of-the-art solutions like deep convolutional neural networks computationally. So, our solution has added a valuable addition to touchless gesture-based technologies to control the computer. This can also be used to recognize gestures, which can also help deaf and mute people.

V. Future Work

In the future, the number of signs can be increased to enhance the vocabulary, which will broaden the spectrum of communication with technology like computers and interpersonal communication. The dynamic signs, which involve movements of hands or other body parts, can be integrated for more nuanced interaction. Continuous learning makes The classifier more adaptable according to user experience. The real-time feedback mechanism can be introduced for constant learning. To increase its accessibility, the solution can be employed on different platforms like Android apps, Raspberry Pi, etc. Some specific movement-based signs can be incorporated by collaboration with healthcare professionals for using this solution for rehabilitation purposes. For example, the rehabilitation of post-stroke survivors involved some movements of hands(these movements can be signs). By the integration of virtual reality or augmented reality, the user experience for rehabilitation or other applications can be increased. So, detection of these movements broadens the spectrum of applications.

References

[1] World Health Organization, World Report On Hearing. 2021. [Online]. Available: https://www.who.int/publications/i/item/world-report-on-hearing

[2] World Health Organisation, World report on vision, vol. 214, no. 14. 2019. [Online]. Available: https://www.who.int/publications-detail/world-report-on-vision

[3] A. Karmel, A. Sharma, M. Pandya, and D. Garg, "IoT based Assistive Device for Deaf, Dumb and Blind People," Procedia Comput. Sci., vol. 165, pp. 259–269, 2019, doi: 10.1016/j.procs.2020.01.080.

[4] S. D. Fisher, "Sign languages in their historical context," Routledge Handb. Hist. Linguist., no. January, pp. 442–465, 2014, doi: 10.4324/9781315794013.ch20.

[5] R. Kumar, S. K. Singh, A. Bajpai, and A. Sinha, "Mediapipe and CNNs for Real-Time ASL Gesture Recognition".

[6] M. Papatsimouli, P. Sarigiannidis, and G. F. Fragulis, "A Survey of Advancements in Real-Time Sign Language Translators: Integration with IoT Technology," Technologies, vol. 11, no. 4, 2023, doi: 10.3390/technologies11040083.

[7] J. Rekha, J. Bhattacharya, and S. Majumder, "Shape, texture and local movement hand gesture features for indian sign language recognition," TISC 2011 - Proc. 3rd Int. Conf. Trendz Inf. Sci. Comput., pp. 30–35, 2011, doi: 10.1109/TISC.2011.6169079.

[8] R. Sutjiadi, "Android-Based Application for Real-Time Indonesian Sign Language Recognition Using Convolutional Neural Network," vol. 12, no. 3, pp. 1541–1549, 2023, doi: 10.18421/TEM123.

[9] J. Bora, S. Dehingia, A. Boruah, A. A. Chetia, and D. Gogoi, "Real-time Assamese Sign Language Recognition using MediaPipe and Deep Learning," Procedia Comput. Sci., vol. 218, no. 2022, pp. 1384–1393, 2023, doi: 10.1016/j.procs.2023.01.117.

[10] Y. Gu, Sherrine, W. Wei, X. Li, J. Yuan, and M. Todoh, "American Sign Language Alphabet Recognition Using Inertial Motion Capture System with Deep Learning," Inventions, vol. 7, no. 4, pp. 1–15, 2022, doi: 10.3390/inventions7040112.

[11] A. M. Buttar, U. Ahmad, A. H. Gumaei, A. Assiri, and M. A. Akbar, "Deep Learning in Sign Language Recognition : A Hybrid Approach for the Recognition of Static and Dynamic Signs," pp. 1–20, 2023.

[12] F. Morillas-Espejo and E. Martinez-Martin, "Sign4all: A Low-Cost Application for Deaf People Communication," IEEE Access, vol. 11, pp. 98776–98786, 2023, doi: 10.1109/access.2023.3312636.

[13] M. Rinalduzzi, De Angelis, A., Santoni, F., Buchicchio, E., Moschitta, A., Carbone, P., Bellitti, P. and Serpelloni, M., "Gesture recognition of sign language alphabet using a magnetic positioning system," Appl. Sci., vol. 11, no. 12, 2021, doi: 10.3390/app11125594.

[14] K. Bhat and C. L. Chayalakshmi, "Advanced Glove for Deaf and Dumb with Speech and Text Message on Android Cell Phone," 2020 IEEE Int. Conf. Innov. Technol. INOCON 2020, pp. 1–7, 2020, doi: 10.1109/INOCON50539.2020.9298283.

[15] A. Dogra, K. Malik, and V. Chowdary, "Sign Language Interpreter," Int. J. Eng. Technol., vol. 7, no. 3.12, p. 990, 2018, doi: 10.14419/ijet.v7i3.12.17619.

[16] M. Shahabuddin, M. Nur Uddin, J. I. Chowdhury, S. F. Ahmed, M. N. Uddin, M. Mofijur, and M. A. Uddin, "A review of the recent development, challenges, and opportunities of electronic waste (e-waste)," Int. J. Environ. Sci. Technol., vol. 20, no. 4, pp. 4513–4520, 2023, doi: 10.1007/s13762-022-04274-w.

[17] A. Roy, A. Saji, M. S. Gokul, and S. Surendran, "A Virtual Assistant for the Visually Impaired," Lect. Notes Electr. Eng., vol. 881, pp. 49–57, 2022, doi: 10.1007/978-981-19-1111-8_5.

[18] A. KARTHIK, V. K. RAJA, and S. PRABAKARAN, "Voice Assistance for Visually Impaired People," in 2018 International Conference on Communication, Computing and Internet of Things (IC3IoT), 2018, pp. 465–468. doi: 10.1109/IC3IoT.2018.8668188.

[19] S. Sayago, B. B. Neves, and B. R. Cowan, "Voice assistants and older people: Some open issues," ACM Int. Conf. Proceeding Ser., pp. 22–24, 2019, doi: 10.1145/3342775.3342803.

[20] G. Germanos, D. Kavallieros, N. Kolokotronis, and N. Georgiou, "Privacy Issues in Voice Assistant Ecosystems," in 2020 IEEE World Congress on Services (SERVICES), 2020, pp. 205–212. doi: 10.1109/SERVICES48979.2020.00050.

[21] N. K. Dim and X. Ren, "Designing Motion Gesture Interfaces in Mobile Phones for Blind People," J. Comput. Sci. Technol., vol. 29, no. 5, pp. 812–824, 2014, doi: 10.1007/s11390-014-1470-5.