Real-time Dynamic Sign Recognition using MediaPipe

Youssef FARHAN
Advanced Systems Engineering Laboratory
National Schools of Applied Sciences, Ibn Tofail University
Kenitra, Morocco
youssef.farhan@uit.ac.ma

Abdessalam AIT MADI
Advanced Systems Engineering Laboratory
National Schools of Applied Sciences, Ibn Tofail University
Kenitra, Morocco
abdessalam.aitmadi@uit.ac.ma

2022 IEEE 3rd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS) | 978-1-6654-5723-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICECOCS55148.2022.9982822

***Abstract*— People live in societies, so communication is necessary to express their needs and ideas. Deaf-mute people use sign language as their means of communication, but most people are unaware of its meaning, so creating an effective system to help them communicate is important. This paper proposes an American Sign Language (ASL) recognition system for 12 dynamic signs in real-time using the MediaPipe framework and a Long Short-Term Memory (LSTM) network. To improve system performance, only the relevant features were extracted from the dynamic sign video, and two new useful features were generated (angles and distances). The proposed system achieved a test accuracy of 97,2%.**

Keywords—American Sign Language, dynamic signs, MediaPipe, LSTM, computer vision

I. INTRODUCTION

Communication is one of the most important pillars in daily life since it allows people to share their ideas and express their opinions, enabling them to integrate into society. However, there is a group of people who are deprived of the blessing of hearing and speech, or they find it difficult to use it, which means that they are unable to communicate normally, causing them to have a problem integrating into society.

Here comes the role of sign language, which greatly contributes to the improvement of the lives of deaf-mute people by allowing them to communicate with society and share ideas, information, and emotions with others. Despite the importance of sign language, a societal gap hinders the communication of deaf-mute people with others. The reason behind this is that deaf-mute people are unable to express their thoughts or receive the ideas of others, due to the lack of knowledge of most people of this language. Consequently, a solution that facilitates communication for this group of people must be made to provide these deaf-mute persons with the opportunity to integrate into society. This solution may be used to instantly translate signs into text like our previous work [1].

Many sign languages are used around the world, including Chinese, French, Arabic, Russian, and American. With such a large number and variety of sign languages, the system for identifying signs remains a major challenge. Moreover, since the most widely used sign language in the world is American Sign Language (ASL), creating an automatic translation system for this language is a reasonable choice.

The two types of ASL recognition are static and dynamic signs [2]. The static signs are easier to identify and translate, but the dynamic ones are not. They are dependent on movement and hence difficult to detect because recognizing them requires following all sign movements for a large number of frames and then processing every one of these frames before recognizing the sign.

The proposed paper focuses on the real-time identification and translation of dynamic signs.

During the execution of the dynamic sign, a large number of sequential information and features must be extracted and analyzed, based on which the sign can be identified. What complicates the situation is the presence of many information. That isn't useful for identifying the sign, such as background, skin tone, and colors, which will have a negative impact on the model's accuracy on the one hand, and will take a long time and effort to train on the other hand. That's why it was decided to extract only the features that are useful for dynamic sign recognition and discard everything else.

To implement this process, the MediaPipe framework was used, which was created specifically for determining the coordinates of various points on the human body, followed by the build and training of a Long Short-Term Memory (LSTM) model.

The rest of the paper is organized as follows. The second section presents the related work. The third section investigates the proposed work. The fourth section presents and discusses the obtained results, and the last section concludes this paper.

II. RELATED WORKS

Many studies continue to struggle with dynamic sign recognition. Researchers have attempted to overcome the difficulties related to ASL recognition in a variety of ways since the 1980s [3]. The majority of approaches for recognizing sign movements rely on vision, sensor gloves, and color.

One of the first techniques, which has been revisited numerous times over the years, is based on the use of a sensor glove capable of tracking hand movements [4], [5]. However, this solution has the disadvantage of requiring hardware. Also, the gloves are typically large and uncomfortable, and the overall cost of all the sensors is too pricey.

The use of simply RGB channels by employing the Hidden Markov Model (HMM), which is used to deal with sequential data from dynamic Sign Language [6], for sign identification has already been attempted by several researchers since 1997, one of the most recent being Kumar [7]. An interesting study [8] has been done claiming that a model based on LSTM

978-1-6654-5723-1/22/$31.00 ©2022 IEEE

performs better than HMM. Therefore recently, Recurrent neural networks (RNN) and LSTM have taken the position of the HMM in dealing with sequential data [9].

In recent years, there has been a growing interest in feature extraction with neural networks due to their greater capabilities. Many studies have combined CNN and LSTM models for dynamic sign recognition tasks [10]–[12], using the CNN models for feature extraction while LSTM models are used for dynamic sign classification.

More recently, the MediaPipe framework has been applied in dynamic sign recognition tasks, where it is capable of properly recognizing human body parts. This framework was used to extract landmark coordinates before training them with various neural network models, such as RNN [9], [13], [14], LSTM [15], and 1D CNN [16]. Furthermore, the extracted landmark coordinates from the MediaPipe framework were used to generate distances and angles [17], but only for dealing with static signs. Table 1 below presents an overview of all these works by summarizing their achievements and limitations.

The approaches described above have several drawbacks for accurately identifying dynamic signs, such as the requirement of large datasets for the CNN-based model and a complicated process for the glove-based or HMM-based models. Real-image-based models are affected by the background, skin tone, colors, and/or clothing style, which is another drawback. Even when MediaPipe is used but no useful features are generated, the method achieves modest results. Moreover, a shorter training time can make a difference because it makes it possible to change the model's hyper-parameters many times to achieve desirable results, which is difficult to achieve when using a complex model with a large number of features that require a long learning time. Therefore, the MediaPipe framework was used in this paper to extract only useful features such as landmark coordinates related to dynamic signs. To improve the proposed approach, two new features (angles and distances) were created based on the extracted coordinates, ensuring that the model does not require a large dataset, and signs recognition task is unaffected by the background, skin tone, colors, or clothing style.

TABLE I. A SUMMARY OF THE INVITED RELATED WORKS

Achievement	Limitation
sensor gloves-based system	
A system based on a sensor glove to translate the sign language gestures into sentences in the English language.	- inaccessible and ungeneralizable due to reliance on external hardware. - the gloves are often big and uncomfortable. - the total cost of all the sensors is too expensive.
An approach based on the Hidden Markov Model (HMM)	
A system for recognizing Sign Language using HMM.	- Traditional computer vision approach. - The negative of HMM is memory loss.
combined CNN and LSTM	
A system for dynamic hand gesture recognition combines CNN for feature extraction and an LSTM network for input gesture prediction.	- need a huge dataset - Training takes a long time. - influenced by the environment, skin tone, or colors.
Combined MediaPipe and neural network	
A system based on the implementation of machine learning methods (LSTM, RNN, 1D CNN) with the Mediapipe framework as a feature extractor.	- No significant features are being generated. - Despite the addition of new features, dealing with static signs.

III. PROPOSED WORK

A. Proposed training process workflow

Figure 1 provides an overview of the model's development and training process. Several stages were completed to build the model which can be summarized as video filming of dynamic signs, extraction of custom landmark coordinates using MediaPipe, calculating angles and distances, storing all these features in a Numpy array, and using the data stored in the array as data to train the LSTM network to obtain a trained model.

B. Proposed testing process

After training the model, the testing phase begins. During this phase, the model is tested in two stages: one using test data to determine the model performance metrics and the number of true and false predictions, and another using a computer camera to examine the response and accuracy of the model while being tested in real situations. These stages are depicted in figure 2.

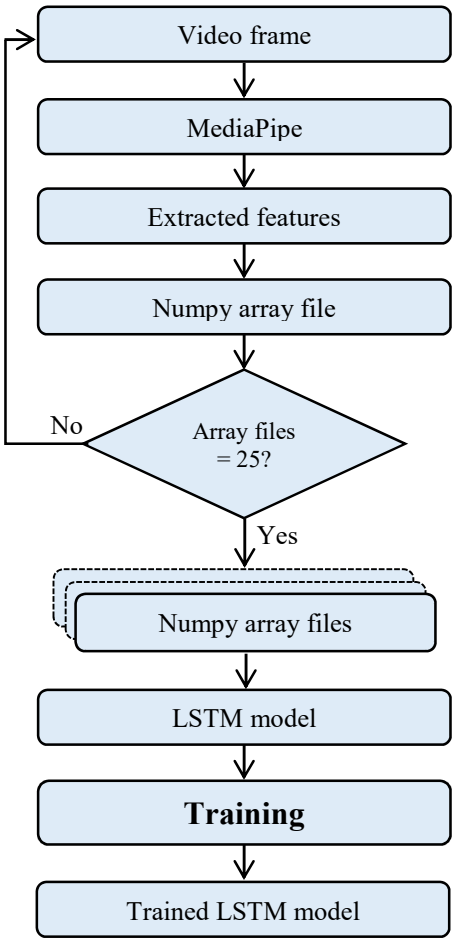

Fig. 1. Training process

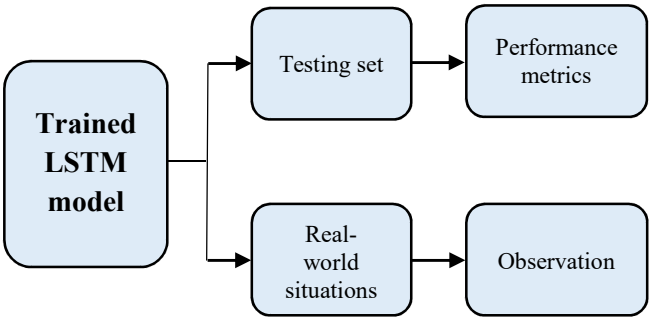

Fig. 2. Testing process

C. Proposed Model Architecture

Before recognizing the sign's name, it is necessary to track multiple frame features, 25 in this case, to identify the sign. To put it another way, sequential data will be processed 25 times, group by group, with 285 features in each group. As a result of this, the LSTM network, a more advanced version of RNN that excels at handling sequential data, is used.

A network was built consisting of three LSTM layers, then three Dense layers, and finally the output layer consisting of 12 neurons, which is the same number of dynamic signs to be identified. Figure 3 depicts the network's layers as well as the number of parameters to be trained.

Aside from the training parameters, additional adjustments referred to as hyper-parameters must be made before training, and the model cannot indeed work without them. A hyper-parameter is a factor whose value is used to control the learning process. In consideration of these hyper-parameters, the learning algorithm derives the parameters from the data. Their settings influence the training results. Table 2 shows the final hyper-parameters that were specified for this proposed model.

D. Dataset

Creating and using the training and testing datasets is an important aspect of the Artificial Intelligence (AI) field. In this subsection, we will describe the process of collecting datasets, which included three major phases:

1) Primarily dataset

Here, a short video was filmed for each dynamic sign. Each video contains 25 frames; it was noticed that 25 frames are a good number to capture the entire sign movement; when this number is increased, the video remains recorded despite the sign being completed, and when this number is reduced, the video is trimmed off before the sign movement is completed, so 25 frames were chosen for each sign.

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 25, 64)	89600
lstm_1 (LSTM)	(None, 25, 128)	98816
lstm_2 (LSTM)	(None, 25, 64)	49408
lstm_3 (LSTM)	(None, 64)	33024
dense (Dense)	(None, 64)	4160
dense_1 (Dense)	(None, 32)	2080
dense_2 (Dense)	(None, 16)	528
dense_3 (Dense)	(None, 12)	204

Total params: 277,820
Trainable params: 277,820
Non-trainable params: 0

Fig. 3. The model summary

TABLE II. THE FINAL HYPER-PARAMETERS

hyper-parameters	Value
Training data	80% (48 video)
Testing data	20% (12 video)
Sequence length	25
Hidden layers + neurons per layers	LSTM 128 neurons LSTM 64 neurons LSTM 64 neurons Dense 64 neurons Dense 32 neurons Dense 32 neurons
Activation function – input and hidden layers	Relu
Activition function - output layer	Softmax
Dropout	No
Regularization	No
Optimizer	Adam
Batch size	32 (default value)
Epoch	300

The purpose of the work done in this article was to identify 12 different dynamic signs commonly used in daily conversation, using an AI model. These dynamic signs are: "Hello", "Thanks", "Yes", "No", "Father", "Mother", "Fine", "Please", "Forget", "Busy", "Happy", "Sad". For each dynamic sign, 60 videos were recorded, which means 720 videos.

2) Features engineering

In this step, only useful features from the video of the dynamic sign are extracted, which are sufficient to identify the sign's name. When a video of a dynamic sign is recorded, it contains a wealth of information. Some of them are useful for identifying signs, such as hand movement, finger position, and the distance between the hand and the face. However, many of them are not, such as skin color, background, and clothing type. These last features, despite their abundance, are ineffective for recognizing dynamic signs.

Therefore, the useless information was discarded, leaving only the useful information to be used in recognizing dynamic signs. This is beneficial in two ways: it improves the model's accuracy by focusing only on the information necessary for it to recognize the sign, and it increases training speed by removing many unnecessary details.

For this purpose, the MediaPipe framework was used, which was specially developed to determine the coordinates of specific points in the human body, the Figures 4 and 5 illustrate the detected landmarks using the hand model and pose model, respectively. Because sign language is more than just hand movement, the Holistic model was used, which

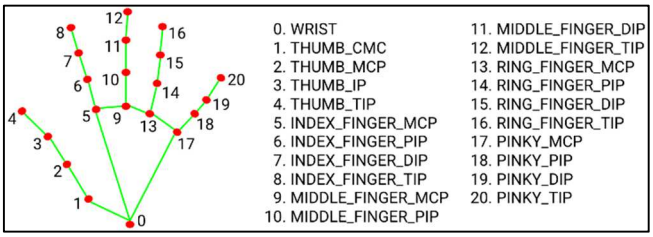

Fig. 4. Landmarks of hand model [19]

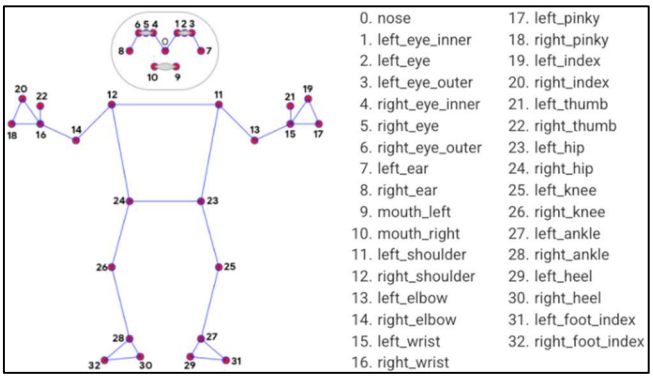

Fig. 5. Landmarks of pose model [20]

combines three models at the same time to determine the coordinates of landmarks on the face, hand, and pose [18].

Since the features of the face do not differ significantly from one sign to another, the face's landmarks from the face model, which number 468 in total, have been abandoned, as have the landmarks below the body; from the waist to the feet. In addition to the full hand's landmarks kept from the hand model, only the landmarks of the arms, hands, thickens, and face were retained from the pose model. Thus, only 65 landmarks are kept, which are 23 and 42 landmarks retained, respectively, from pose and hand (left and right) models.

Thereby, the coordinates of these landmarks were extracted using MediaPipe and newly created functions, with four values for the pose landmarks (x, y, z, and visibility) and three values for the hand landmarks (x, y, z), for a total of 218 coordinate values extracted.

The coordinates of the landmarks are insufficient for correctly identifying dynamic signs. As a result, it should be supplemented with additional information for the dynamic sign to be easily identified. For this purpose, angles and distances are two new features that have been added based on the extracted coordinates.

When making a dynamic sign, some parts of the body are stretched or bent; in other words, the angles of the body's landmarks vary depending on the sign. As a result, this concept can be used to exploit the change in angles by calculating it, providing a clearer idea of the sign. Using the x and y coordinates of three landmarks, the angles were calculated using the mathematical relations given by equations (1) and (2).

$$\widehat{ABC}_{rad} = \text{atan2}(y_c - y_B, x_c - x_B) - \text{atan2}(y_A - y_B, x_A - x_B) \quad (1)$$

$$\widehat{ABC}_{deg} = \widehat{ABC}_{Rad} \times \frac{180}{\pi} \quad (2)$$

The trio landmarks used in this paper are shown in table 3, which gives a total of 41 angle values.

TABLE III. LANDMARKS USED FOR ANGLE CALCULATING

Angles Pose		Angles Hands	
Angle number	*Landmarks*	*Angle number*	*Landmarks*
1	[1, 0, 4]	1	[2, 3, 4]
2	[1, 3, 7]	2	[5, 6, 7]
3	[4, 6, 8]	3	[6, 7, 8]
4	[11, 12, 14]	4	[9, 10, 11]
5	[12, 11, 13]	5	[10, 11, 12]
6	[12, 14, 16]	6	[13, 14, 15]
7	[11, 13, 15]	7	[14, 15, 16]
8	[22, 16, 18]	8	[17, 18, 19]
9	[21, 15, 17]	9	[18, 19, 20]
10	[19, 11, 7]	10	[4, 0, 8]
11	[20, 12, 8]	11	[8, 0, 20]
		12	[16, 17, 20]
		13	[8, 5, 12]
		14	[4, 5, 20]
		15	[8, 13, 20]

Given that the angle can take values ranging from -180 to 180, which are large values when compared to the coordinates extracted using MediaPipe, which take values ranging from 0 to 1 [18], it should be normalized to take values close to the values of the extracted coordinates from MediaPipe. And, for this purpose, the angles' values have been divided by 180, so the angles' values are limited to -1 to 1.

As for the distances, the distance between one landmark and another varies during the performance of the dynamic sign, as it differs between one sign and another, and as a result, extracting the distance between some landmarks is a useful feature for recognizing the name of the sign. The distance between two points was calculated using their coordinates (x, y) by the mathematical relation given by equation (3).

$$d_{AB} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} \quad (3)$$

The doublet landmarks used for calculating distance are shown in table 4, which gives a total of 26 distance values.

3) Dataset collection

Finally, all these features, coordinates of custom landmarks, angles, and distances, were collected, using the Numpy library, in a single array file. It contains only useful features for recognizing dynamic signs and contains no information or useless features. In this matter, the generated Numpy arrays can be much better than preserving the entire video, as it contains just the most useful features. In total, 285 features are stored in each array file. The above operations are summarized in figure 6.

All of these steps, including video filming of dynamic signs, identifying custom landmarks, extracting their coordinates, calculating angles and distances, and collecting and storing these features in a single file, were accomplished

TABLE IV. LANDMARKS USED FOR DISTANCE CALCULATING

Distances Pose		Distances Hands	
Distance number	*Landmarks*	*Distance number*	*Landmarks*
1	[0, 7]	1	[0, 4]
2	[0, 8]	2	[0, 8]
3	[0, 20]	3	[0, 12]
4	[0, 19]	4	[0, 16]
5	[7, 8]	5	[0, 20]
6	[12, 20]	6	[4, 8]
7	[11, 19]	7	[4, 20]
8	[11, 12]	8	[8, 20]
9	[15, 16]		
10	[19, 20]		

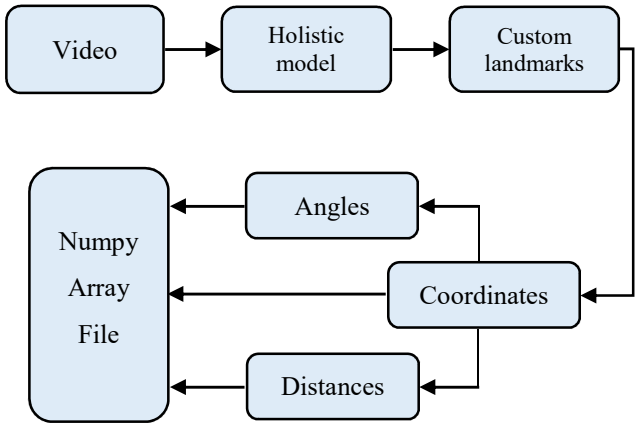

Fig. 6. Steps of dataset collection

in real time while filming a video for each dynamic sign. In another word, before capturing the next frame, the previous frame's features will be stored in Numpy array form.

IV. Results and Discussion

To obtain results, the LSTM network designed to deal with serial data has been trained to classify 12 dynamic signs, as it contains 277820 parameters that will be trained using the training data and adjusted using the optimizer and loss functions.

The model was trained 300 times, and during the training, the loss value and the categorical_accuracy value were recorded for each stage. Then, the model was tested using test data, and in real time with a standard computer camera.

A. Simulation results

Figures 7 and 8 display the results recorded during the training. Figure 7 depicts the loss variation over the training epoch; the graph shows that the loss value began greater than 0.7 at the start of the training and gradually decreased fluctuating until the training reached epoch 160, at which point the loss value decreased significantly until it reached the value of 6.85e-6 at the end of the training. The same can be stated in Figure 8, which depicts categorical_accuracy variation during training, as it began with a value of less than

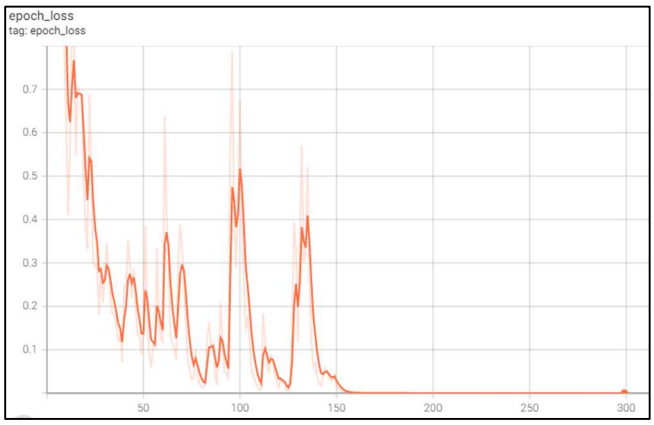

Fig. 7. Loss value during training

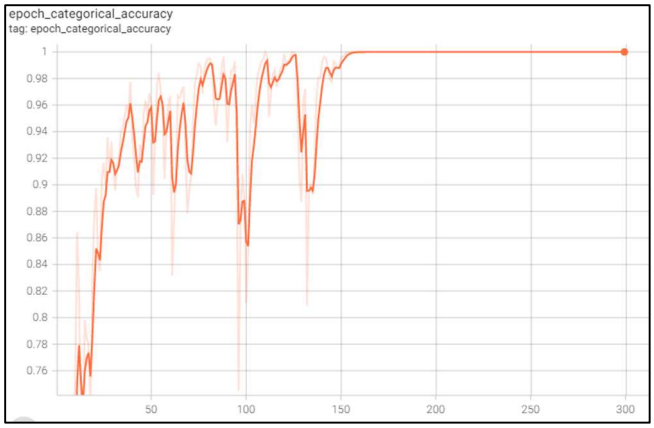

Fig. 8. Accuracy value during training

76% and gradually increased in a fluctuating manner until training reached epoch 160, at which point the categorical_accuracy value stabilized at 100% until the end of training.

The training phase took less than 4 minutes, which is a very short time, compared to the time generally needed to train neural network models. This is due to the reduction of the features and information in the video's frame, where only the important information to recognize the sign was kept. This short time allows for adjusting the model's hyper-parameters after the training is finished if the results are not desired, allowing the possibility of repeating the training with different hyper-parameters to obtain the desired results.

During the training phase, the training accuracy reached 100% after a step of 160 out of 300 to settle at this value after that, which is a good indicator, while the loss value continued to decrease gradually until it finally reached 6.85e-6, a very small value, which proves that the model was trained excellently.

Following the completion of the training, the model was tested using test data to determine the performance metrics by measuring the precision, recall, and f1-score values for each dynamic sign, which are resumed in figure 9, with the support column indicating the number of test images for each sign.

Based on figure 9, the average precision, recall, and f1-score values are 97%, 96,75% and 96,75% respectively. For effective presentation test results, figure 10 depicts the model predictions for each dynamic sign, using a confusion matrix.

As for the accuracy of the test, it was achieved at 97,2%,

	precision	recall	f1-score	support
Hello	1.00	1.00	1.00	14
Thanks	1.00	0.78	0.88	9
Yes	0.89	1.00	0.94	8
No	1.00	1.00	1.00	12
Father	1.00	1.00	1.00	14
Mother	0.92	1.00	0.96	11
Fine	0.92	0.92	0.92	13
Please	0.91	0.91	0.91	11
Forget	1.00	1.00	1.00	19
Busy	1.00	1.00	1.00	14
Happy	1.00	1.00	1.00	10
Sad	1.00	1.00	1.00	9

Fig. 9. Performance evaluation

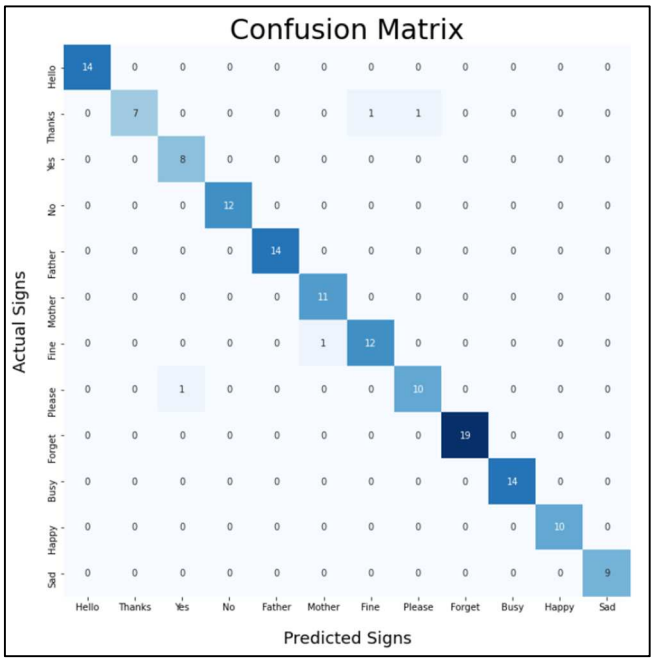

Fig. 10. Confusion matrix

which is a good percentage because of training the model on 12 dynamic signs, which makes it difficult to reach high accuracy, but the proposed model achieved this good percentage due to the disposal of all useless features such as colors and backgrounds, and the extraction of only important features, which are the coordinates of the landmarks of the hands and the upper part of the pose using MediaPipe, and creating two useful features for recognizing the dynamic signs are distances and angles, these two features played a significant role in enabling the model to accurately classify the signs. Thus, extracting 285 important features from the video frame was critical in achieving the model's high accuracy while also significantly reducing training time.

B. Real-Time results

To determine the model's accuracy in real-world cases of dynamic signs, the model was tested in real time with a standard computer camera. To determine the accuracy of sign recognition, a bar has been added to the screen, the bar high changes by changing the sign detection score from 0 to 100 percent, so that the score of dynamic sign detection can be seen visually in real time, and due to a lack of space to write names, the names of the signs in this part were replaced with equivalence numbers; 1 for "Hello", 2 for "Thanks", 3 for "Yes", 4 for "No", 5 for "Father", 6 for "Mother", 7 for "Fine", 8 for "Please", 9 for "Forget", 10 for "Busy", 11 for "Happy", 12 for "Sad". To determine whether the prediction is correct or not, the name of the detected sign is displayed in real time in a band that has been added at the top of the screen, in a way that the name of the currently detected sign is written at the end of the list. Figure 11 shows some examples of testing in real-world situations.

Figure 11 gives examples of some dynamic signs that are being tested in real time using a standard computer camera. For example, the image above on the left depicts the start of the "Forget" movement, and it appears that the model correctly recognized it by writing its name at the end of the list of names on the bar at the top. In addition to correctly translating it, the test demonstrates that the model accurately recognized the dynamic sign "Forget" via the height changeable bar that appears at the bottom of the image at number 9, which is the number equivalent to the "Forget" sign, as the detection score refers to 100 while it refers to 0 for the rest of the signs, which indicates the great accuracy of the model. The same may be said for the other three images.

So in general, during the test in real-world situations, the model correctly recognized the dynamic signs and wrote their names in the bar designated for this purpose. Moreover, it was noted that the current sign has a high detection score, which is shown by the height changeable bar. This score is considerable when compared to the scores of the rest of the signs, indicating that the model recognizes the current sign accurately. When the transition from one sign to another occurs, the response speed is somewhat slow, because the model takes some time to recognize the new dynamic sign, implying that the model has some limitations when it comes to recognizing successive dynamic signs.

Dynamic sign recognition accuracy is primarily dependent on the MediaPipe framework, and because the latter typically correctly recognizes body landmarks, this has a beneficial impact on model accuracy. However, in some cases, the MediaPipe framework has difficulties identifying the hands' landmarks if they are at an undesirable angle, such as when they are in a horizontal position, therefore, the model suffers when attempting to recognize the sign, but this seldom happens.

Fig. 11. Testing in real situations

V. Conclusion

Because most people do not understand sign language, Deaf-mute people have many difficulties interacting with others. As a result, building a system capable of bridging the barrier is necessary. This paper presented a dynamic sign recognition system based on the MediaPipe framework for extracting custom landmark coordinates. Over this, dynamic sign recognition is unaffected by the image's background, clothing style, or person's skin tone. As previously stated, the trained model, which was based on 12 popular ASL words, achieved a good test accuracy. That is, the LSTM model trained very well on the extracted features and performed well in the testing phase. As a consequence, shortly, the research may include many words based on dynamic signs.

References

[1] Y. Farhan, A. Ait Madi, A. Ryahi, and F. Derwich, "American Sign Language: Detection and Automatic Text Generation," in *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, Mar. 2022, pp. 1–6, doi: 10.1109/iraset52964.2022.9738061.

[2] R. A. Kadhim and M. Khamees, "A real-time american sign language recognition system using convolutional neural network for real datasets," *TEM J.*, vol. 9, no. 3, pp. 937–943, 2020, doi: 10.18421/TEM93-14.

[3] A. Costa, "ASLScribe : Real-Time American Sign Language Alphabet Image Classification Using MediaPipe Hands and Artificial Neural Networks," 2021.

[4] J. Galka, M. Masior, M. Zaborski, and K. Barczewska, "Inertial Motion Sensing Glove for Sign Language Gesture Acquisition and Recognition," *IEEE Sens. J.*, vol. 16, no. 16, pp. 6310–6316, 2016, doi: 10.1109/JSEN.2016.2583542.

[5] S. A. Mehdi and E. Sciences, "Sign Language Recognition using Sensor Gloves Yasir Niaz Khan," *Neural Inf. Process.*, vol. 5, pp. 2204–2206, 2007.

[6] Suharjito, M. C. Ariesta, F. Wiryana, and G. P. Kusuma, "A survey of hand gesture recognition methods in sign language recognition," in *Pertanika Journal of Science and Technology*, vol. 26, no. 4, 2018, pp. 1659–1675.

[7] P. Kumar, H. Gauba, P. P. Roy, and D. P. Dogra, "Coupled HMM-based multi-sensor data fusion for sign language recognition," *Pattern Recognit. Lett.*, vol. 86, pp. 1–8, 2017, doi: 10.1016/j.patrec.2016.12.004.

[8] T. Liu, W. Zhou, and H. Li, "SIGN LANGUAGE RECOGNITION WITH LONG SHORT-TERM MEMORY Tao Liu , Wengang Zhou , and Houqiang Li University of Science and Technology of China Department of Electronic Engineering and Information Science," pp. 2871–2875, 2016.

[9] B. Duy Khuat, D. Thai Phung, H. Thi Thu Pham, A. Ngoc Bui, and S. Tung Ngo, "Vietnamese sign language detection using Mediapipe," in *2021 10th International Conference on Software and Computer Applications*, Feb. 2021, pp. 162–165, doi: 10.1145/3457784.3457810.

[10] F. Obaid, A. Babadi, and A. Yoosofan, "Hand Gesture Recognition in Video Sequences Using Deep Convolutional and Recurrent Neural Networks," *Appl. Comput. Syst.*, vol. 25, no. 1, pp. 57–61, May 2020, doi: 10.2478/acss-2020-0007.

[11] M. Ur Rehman *et al.*, "Dynamic Hand Gesture Recognition Using 3D-CNN and LSTM Networks," *Comput. Mater. Contin.*, vol. 70, no. 3, pp. 4675–4690, 2021, doi: 10.32604/cmc.2022.019586.

[12] W. Zhang, J. Wang, and F. Lan, "Dynamic hand gesture recognition based on short-term sampling neural networks," *IEEE/CAA J. Autom. Sin.*, vol. 8, no. 1, pp. 110–120, Jan. 2021, doi: 10.1109/JAS.2020.1003465.

[13] D. L. Antonio, "Sign Language Recognition ASL Recognition with MediaPipe and Recurrent Neural Networks," FH Aachen University of Applied Sciences, 2020.

[14] A. Chaikaew, K. Somkuan, and T. Yuyen, "Thai Sign Language Recognition: an Application of Deep Neural Network," in *2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering*, Mar. 2021, pp. 128–131, doi: 10.1109/ECTIDAMTNCON51128.2021.9425711.

[15] M. G. Grif and Y. K. Kondratenko, "Development of a software module for recognizing the fingerspelling of the Russian Sign Language based on LSTM," *J. Phys. Conf. Ser.*, vol. 2032, no. 1, pp. 1–7, Oct. 2021, doi: 10.1088/1742-6596/2032/1/012024.

[16] A. Kanodia, P. Singh, D. Rajesh, and G. Malathi, "Indian sign language using holistic pose detection," *Turkish J. Physiother. Rehabil.*, vol. 32, no. 3, pp. 19746–19752, 2021.

[17] J. Shin, A. Matsuoka, M. A. M. Hasan, and A. Y. Srizon, "American Sign Language Alphabet Recognition by Extracting Feature from Hand Pose Estimation," *Sensors*, vol. 21, no. 17, pp. 1–19, Aug. 2021, doi: 10.3390/s21175856.

[18] Google, "Holistic - Mediapipe," 2020. https://google.github.io/mediapipe/solutions/holistic.html (accessed Apr. 08, 2022).

[19] Google, "Hands - Mediapipe," 2020. https://google.github.io/mediapipe/solutions/hands.html (accessed Apr. 08, 2022).

[20] Google, "Pose - MediaPipe," 2020. https://google.github.io/mediapipe/solutions/pose.html (accessed Apr. 08, 2022).