Performance of the Sign Language Recognition System Using the Long Short-Term Memory Network Algorithm

Lakash Maharjan[1] and Chantri Polprasert[1]

Asian Institute of Technology, Pathumthani, 12120, Thailand
chantri@ait.asia

Abstract. In this paper, we investigate the performance of the vision-based American Sign Language (ASL) recognition system employing the Long Short-Term Memory (LSTM) classifier. MediaPipe introduced by Google [15] is used to obtain posture landmarks where a custom dataset has been created and used for the experimental study. Ten sign words were considered from the ASL: hello, iloveyou, thanks, google, internet, jogging, house, book, drink and drive. The proposed system is benchmarked with the classifier employing the k Nearest Neighbor (kNN) with Dynamic Time Warping (DTW). The proposed system yields 92% accuracy comparable to those obtained from the classifier using the kNN with DTW whose accuracy is equal to 75%. In addition, we evaluated the robustness of the proposed system by corrupting the landmarks with zero-mean additive white Gaussian noise whose standard deviation (SD) ranges from 0.001 to 1. The proposed system maintains accuracy beyond 90% when noise SD is less than or equal to 0.15 but exhibits significant performance drop when the noise SD is beyond this value.

Keywords: Sign Language Recognition · MediaPipe · LSTM · Deep Neural Network · Dynamic Time Warping · kNN.

1 Introduction

According to the World Federation of the Deaf, there are over 300 sign languages used by approximately 70 million deaf people around the world [6]. Sign language is a crucial form of communication for millions of people around the world, particularly for the deaf and hard of hearing communities. It serves as a powerful tool that bridges the communication gap, enabling individuals to express their thoughts, needs, and emotions effectively. Beyond its role in daily interactions, sign language fosters social inclusion, ensuring that deaf individuals can participate fully in society. It is also integral to education, providing deaf students with the means to access information and engage in learning. Recognizing and promoting the importance of sign language is essential in advancing equal opportunities and enriching the cultural and linguistic diversity of our global community. However, sign language users face several challenges. For example,

only a small percentage of the population uses and practices sign language, creating a divide between deaf individuals and those who want to communicate with them. This communication barrier can lead to difficulties in performing simple tasks and impact emotional, mental, and societal development [5]. Many communication technologies and tools have been developed to facilitate communications between deaf people and hearing worlds such as iCommunicator [12], Hand Talk [10] and some wearable electronics [16]. However, challenges remain in ensuring reliability of the technologies under the diverse surroundings, especially the vision-based translation system, which is prone to error in noise corrupted environments [13].

This paper investigates the performance of a vision-based ASL recognition system using an LSTM classifier. MediaPipe by Google [15] was used to extract posture landmarks from a custom dataset with 10 ASL signs: hello, iloveyou, thanks, google, internet, jogging, house, book, drive, and drink. The system achieves 92% accuracy, compared to 75% for the kNN with DTW classifier. The robustness was tested by adding white Gaussian noise to landmarks, and the system maintained over 90% accuracy for noise levels with SD ≤ 0.15.

Section 2 describes a summary of related work. Datasets used to train the detection models are discussed in Section 3. Section 4 explains the proposed system framework, data extraction and pre-processing, and classifier models. Results and discussions of the proposed system are explained in Section 5. Finally, Section 6 describes conclusions and future works.

2 Related Works

Most vision-based sign language recognition (SLR) system involve interpreting hand-shape, palm orientation, movement, location and expression/non-manual signal. Different approaches have been proposed to improve the performance of the SLR system in real-time scenarios. Traditional approaches involve classifying sign language vdo using the kNN [4], support vector machine (SVM) [14], hidden Markov model (HMM) [3] and ensemble learning [9]. With the arrival of deep learning methods, many researchers have investigated the performance of deep learning on SLR tasks. A real-time ASL recognition system using the convolutional neural network (CNN) was proposed [17]. The proposed CNN model consists of the input layer, two 2D convolution layers, pooling, flattening and two dense layers. An accuracy of 98.05% was obtained when tested with the real-time dataset. Authors [2] introduced a deep learning model based on CNN to recognise static ASL alphabets. The model was tested on the American finger spelling A dataset which contains 24 letters of the ASL alphabet. The developed model performed well on the two datasets with 99.96% recognition accuracy. LSTM-based neural network [1] has been used for the SLR of the Indian sign language with accuracy of 98% for 26 gestures. Another approach [11] incorporates a CNN and LSTM for SLR and achieves a recognition accuracy of 91.1% with data augmentation on testing datasets.

3 Datasets

For our study, we use a custom ASL dataset recorded from the computer's webcam with resolution 720p with frame rate equal to 25 fps. Details of the dataset are listed in JSON format illustrating the video IDs, frame rates, and words. 'hello', 'iloveyou', 'thanks', 'google', 'internet', 'jogging', 'house', 'book', 'drink' and 'drive' are selected for the experiment due to its widespread usage in daily communications. The signer performed signs when sitting on a chair in front of a computer laptop on the table. Each word consists of 20 sign videos where each sign is recorded for 1.2 seconds at 25 frame rate. Train and test datasets are randomly split by 50:50.

4 Methodology

4.1 Data Extraction using MediaPipe

In this part of our methodology, we detail the steps from detecting human poses in a video to extracting keypoints, and finally organizing these keypoints in a designated data folder. Estimating human body pose and capturing landmarks are crucial for sign language recognition. Initially, our method involves the detection of keypoints using computer vision techniques. We employ the MediaPipe and OpenCV libraries for real-time detection of human body keypoints in videos from our custom dataset.

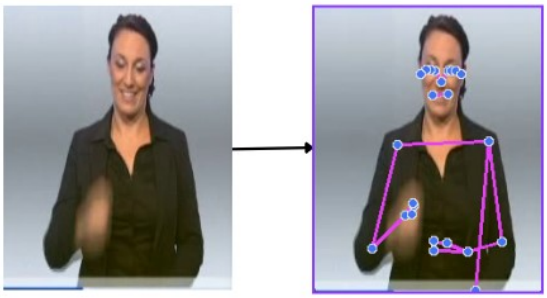

Fig. 1. Showcasing how key point detection works using MediaPipe on the custom dataset [15]

A critical aspect of the SLR is pose estimation, which entails the precise detection and recording of keypoints from an individual's movements in video data. These keypoints, located at vital body parts such as joints and facial areas, are essential for decoding the subtleties of sign language gestures. For effective pose estimation, we utilize Google's MediaPipe Holistic library [15]. This tool allows for an integrated approach to estimating pose by simultaneously analyzing facial

landmarks, hand gestures, and body posture. Such comprehensive processing not only improves the accuracy of our sign language recognition model but also ensures thorough capture and interpretation of the gesture nuances, providing a robust basis for further analysis. Utilization of MidiaPipe on the custom dataset is shown in Fig. 1.

Using the MediaPipe library on our video data, we extract an extensive set of 1662 three-dimensional keypoints (x, y, and z coordinates) from a single frame. This includes 21 keypoints for each hand, as illustrated in Fig. 2, which captures detailed finger and palm movements; 33 keypoints from the body, providing a full view of the core and limbs to assess posture; and 468 keypoints from the face, accurately documenting facial expressions. This comprehensive collection of keypoints offers a nuanced representation of the individual's movements.

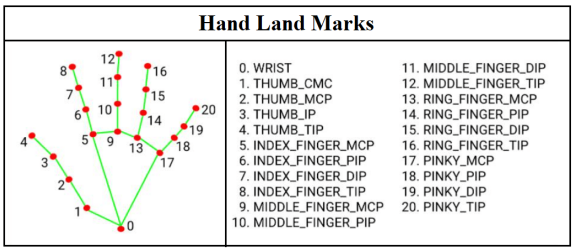

Fig. 2. Hand Landmarks from MediaPipe [15]

Using three-dimensional coordinates (x, y, and z), the calculation of extracted keypoints is as follows:

For the hands:

$$\text{Keypoints in hand} \times \text{Dimensions} \times \text{Number of hands} = 21 \times 3 \times 2 = 126$$

For the body pose (including visibility):

$$\text{Keypoints in pose} \times (\text{Dimensions} + \text{Visibility}) = 33 \times (3 + 1) = 132$$

For the face:

$$\text{Keypoints in face} \times \text{Dimensions} = 468 \times 3 = 1404$$

This comprehensive dataset is pivotal to our analysis, offering a detailed understanding of sign language gestures through the capture of human movements in three-dimensional space.

MediaPipe outputs a 3D array where each inner array represents a single frame's landmarks. Each landmark, such as hand key points or body joints, is described by three values: normalized X and Y coordinates for position and a Z coordinate for depth. This setup captures 3D spatial data for each landmark. Keypoint extraction is shown in Fig. 3.

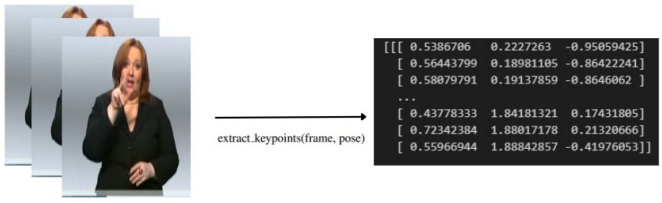

Fig. 3. Extracting pose keypoints from video frames using a MedaiPipe Holistic. We took for Phoenix 2014T dataset to demonstrate Feature extraction in this figure.

We developed an automated pipeline using OpenCV to extract pose keypoints from live video. Keypoints for each sign are stored in a designated path, compiled into a flattened NumPy array, and saved in .npy format for efficient storage and easy retrieval.

4.2 Data Pre-processing and Partitioning

This section explains steps that involve creating a standardized mapping of sign language gestures to numerical identifiers. This process is important for transforming the vocab into a format that would be easy for model to analyze. To do this, we employ a dictionary, 'label_map', which assigns a unique integer to each gesture label based on its occurrence in a predefined list, vocabulary. This mapping ensures that each sign language gesture, such as 'iloveyou', 'google', 'internet', etc, are represented by a distinct numerical value, facilitating their identification and processing by our model. Such a conversion from textual to numerical representation not only enhances computational efficiency but also ensures consistency across the dataset.

We will consolidate the data into a single structure. For example, with three vocabulary items (a, b, c), we have 20 arrays representing 10 videos per vocab, each containing 30 frames. Each frame has 1662 values representing detected key points. Sequences are added to a sequence list, and corresponding labels from the "label map" are added to a label list, ensuring each sequence has a matching label.

The sequences are converted into a NumPy array X with the shape (number of sequences, sequence length, frame dimension). Labels are stored in y and one-hot encoded using to_categorical. This transforms labels into a binary matrix where rows correspond to sequences and columns to classes. After creating X and y, the dataset is split into training and testing subsets using the train_test_split function from sklearn.model_selection, ensuring balanced training and unbiased performance evaluation on unseen data.

4.3 Classifier

We investigate the performance of the proposed classifier using the LSTM. The proposed algorithm is benchmarked with those using the k-NN and DTW algorithm.

Table 1. Model Summary of Sequential LSTM and Dense Layers

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 30, 64)	442,112
lstm_4 (LSTM)	(None, 30, 128)	98,816
lstm_5 (LSTM)	(None, 64)	49,408
dense_3 (Dense)	(None, 64)	4,160
dense_4 (Dense)	(None, 32)	2,080
dense_5 (Dense)	(None, 10)	330

Total params	596,906 (2.28 MB)
Trainable params	596,906 (2.28 MB)
Non-trainable params	0 (0.00 Byte)

LSTM - The architecture of the proposed sequential neural network is shown in 1 with three LSTM layers and three dense layers, totaling 596,906 trainable parameters. This architecture is used for sequence-to-sequence models, time series prediction, and for training and classifying sequential input data. We used TensorFlow's Keras API to build the model, as it comes with an API for LSTM out of the box.

We use the Adam optimizer and categorical cross-entropy as the loss function to train the model. Evaluation measures are accuracy, precision, and recall, and the work was trained over 200 epochs. A block diagram of the overall architecture is shown in the Fig. 4.

kNN with DTW - DTW is a widely used algorithm for time series analysis. It excels in comparing two time series by aligning their data points based on temporal dynamics rather than just their sequence order. By minimizing the distance between these time series, DTW identifies the optimal alignment, making it particularly useful for time series that exhibit similar patterns but are out of sync or differ in length. DTW measures the distance between two data points A and B, to provide minimum distance, this can be computed using (1),

$$DTW(A, B) = \min \sum_{(i,j)\in P} d(a_i, b_j) \tag{1}$$

where $d(a_i, b_j)$ represents the distance between points a_i from series A and b_j from series B, summed over the optimal alignment path. To implement DTW, key points are extracted from the custom dataset through MediaPipe. Each Video has 30 frames of data and each frame contains 1662 key points. Due to

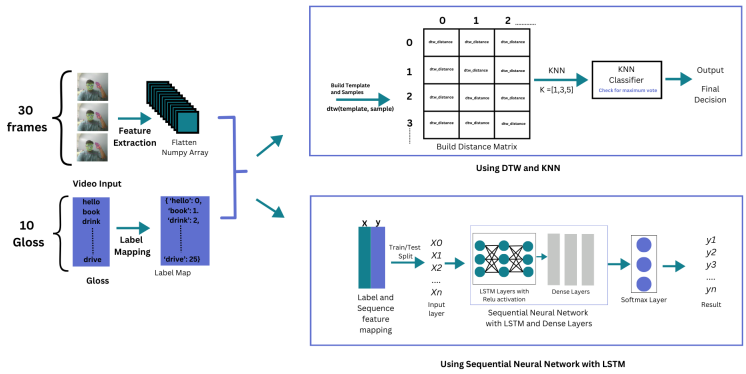

Fig. 4. A block diagram of the proposed system employing the LSTM and KNN with DTW

computational complexity, key points from face are neglected for calculating the DTW. Following the data collection we prepare the data to process dtw calculation. We will be using 'dtaidistance' library to calculate DTW distance. Upon calculating the distance we will form distance matrix, mapping the distance between each gloss. We use the produced matrix to fit the kNN classifier. The distance matrix represents the distance between each pair in the dataset. This matrix allows the classifier to determine how close each point is to all other points without having to recalculate the distance. The classifier is initialized with different K values (k =1,3,5) which will take only the k nearest neighbor and vote for the most likely word.

5 Results and Discussions

Fig. 5 shows the confusion matrix of the proposed SLR employing the LSTM. From the figure, the proposed system exhibits strong performance in translating the sign language. However, the system cannot reliably translate 'iloveyou' sign as it incorrectly interprets this as 'google' for 30% of words. This could be due to similar sign movement between 'iloveyou' and 'google' and need further investigation. Fig. 7 presents the impact of the zero-mean additive white Gaussian noise on the performance of the proposed SLR employing the LSTM. From the figure, the proposed system maintains accuracy above 90% when noise SD is less than or equal to 0.15 but drops significantly when the noise level is beyond this value.

Fig. 6 shows the confusion matrix of the classifier employing the kNN with DTW. he classifier exhibits relatively poor performance, as evidenced by the confusion matrix. Specifically, the classifier frequently misinterprets 'Google' as 'drink,' 'drive' as 'Internet,' and 60% of 'I love you'as 'drink.' This results in

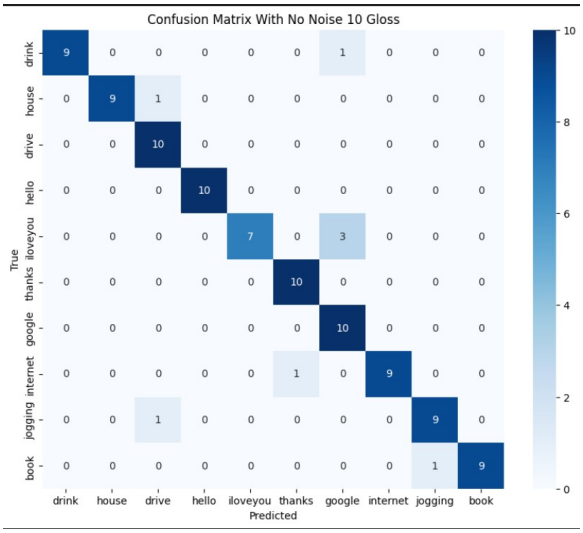

Fig. 5. Confusion matrix of the proposed system without noise.

an overall accuracy of only 75%. These misclassifications are likely due to the similar sign movements between these glosses, which the classifier struggles to differentiate effectively.

Table 2 compares the performance of the proposed system using LSTM with those using the kNN with DTW. From the table, it is evident that the proposed LSTM model exhibits impressive results, significantly outperforming the kNN with the DTW approach. The LSTM model not only achieves higher accuracy but also demonstrates superior robustness in classification tasks. In contrast, the kNN with the DTW approach shows poor performance, particularly struggling with accuracy. Additionally, the kNN with the DTW method requires considerably longer computation time to determine classification results, especially when dealing with large training datasets, further underscoring the efficiency and effectiveness of the LSTM-based model.

Fig. 7 illustrates shows the impact of noise on accuracy for LSTM and DTW&KNN (K=1). LSTM remains robust up to a noise level of 0.1 but then drops sharply in accuracy. DTW&KNN shows a more gradual decline, performing steadily as noise increases. LSTM is ideal for low-noise environments, while DTW&KNN offers more consistent performance under varying noise conditions.

6 Conclusions

We investigate the performance of the vision-based ASL recognition system employing the LSTM classifier. Mediapipe introduced by Google [15] is used to obtain posture landmarks where a custom dataset has been created and used

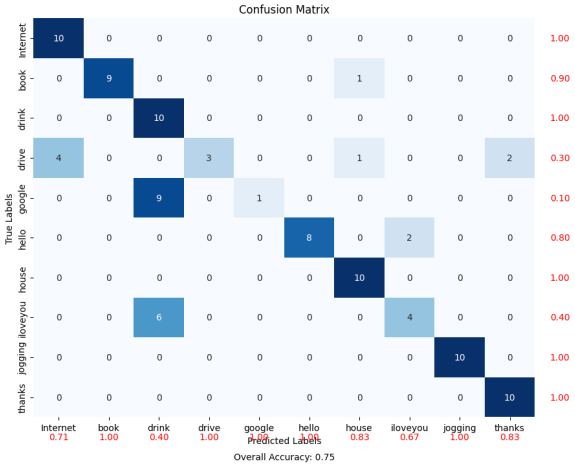

Fig. 6. Confusion matrix of the kNN classifier with DTW, showing correct classifications on the diagonal and misclassifications off-diagonal, without noise on key points.

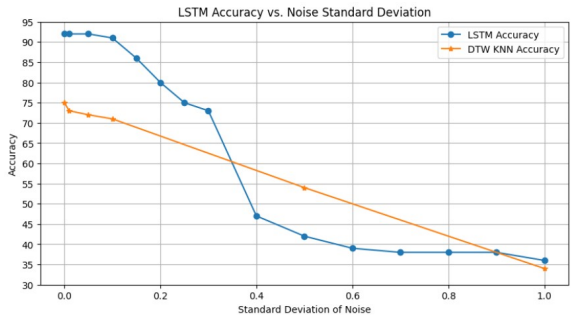

Fig. 7. Performance of the proposed SLR employing the LSTM and KNN with DTW over a range of the zero-mean white Gaussian noise when K=1.

Table 2. Comparison of the accuracy, precision, F1, and recall of the proposed system with that using the kNN with DTW without noise.

Class	Precision	Recall	F1	Accuracy
LSTM	93%	91 %	92%	92%
kNN with DTW (k=1)	70%	60%	56%	75%

for the experimental study. Ten sign words were considered from the ASL: hello, iloveyou, thanks, google, internet, jogging, house, book, drive and drink. The proposed system is benchmarked with the classifier employing the kNN with DTW. The proposed system yields 92% accuracy comparable to those obtained from the classifier using the kNN with DTW whose accuracy is equal to 75%. We evaluated the robustness of the proposed system by corrupting the landmarks

with zero-mean additive white Gaussian noise whose SD ranges from 0.001 to 1. The proposed system maintains accuracy beyond 90% when noise SD is less than or equal to 0.15 but exhibits significant drop when the noise SD is beyond this value. Future works will focus on applying the proposed system on some public sign language datasets such as how2sign [7],RWTH-PHOENIX-2014T [8].

References

1. Abraham, E., Nayak, A., Iqbal, A.: Real-time translation of indian sign language using lstm. In: 2019 global conference for advancement in technology (GCAT). pp. 1–5. IEEE (2019)
2. Adithya, V., Rajesh, R.: A deep convolutional neural network approach for static hand gesture recognition. Procedia computer science **171**, 2353–2361 (2020)
3. Aloysius, N., Geetha, M.: Understanding vision-based continuous sign language recognition. Multimedia Tools and Applications **79**(31), 22177–22209 (2020)
4. Butt, U.M., Husnain, B., Usman, A., Tariq, A., Tariq, I., Butt, M.A., Zia, M.S.: Feature based algorithmic analysis on american sign language dataset. International Journal of Advanced Computer Science and Applications **10**(5) (2019)
5. Pros and Cons of Sign Language/, https://www.educationalwave.com/pros-and-cons-of-sign-language/
6. World Federation of the Deaf, https://wfdeaf.org/
7. Duarte, A., Palaskar, S., Ventura, L., Ghadiyaram, D., DeHaan, K., Metze, F., Torres, J., Giro-i Nieto, X.: How2Sign: A Large-scale Multimodal Dataset for Continuous American Sign Language. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
8. Forster, J., Schmidt, C., Hoyoux, T., Koller, O., Zelle, U., Piater, J., Ney, H.: Rwth-phoenix-weather: A large vocabulary sign language recognition and translation corpus (05 2012)
9. Gupta, R., Jha, N.: Real-time continuous sign language classification using ensemble of windows. In: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). pp. 73–78. IEEE (2020)
10. Hand Talk, https://www.handtalk.me/en/
11. Hernandez, V., Suzuki, T., Venture, G.: Convolutional and recurrent neural network for human activity recognition: Application on american sign language. PloS one **15**(2), e0228869 (2020)
12. icommunicators, https://www.lifewire.com/top-alternative-and-augmentative-communication-198828
13. Jang, Y., Oh, Y., Cho, J.W., Kim, D.J., Chung, J.S., Kweon, I.S.: Signing outside the studio: Benchmarking background robustness for continuous sign language recognition. arXiv preprint arXiv:2211.00448 (2022)
14. Joshi, G., Singh, S., Vig, R.: Taguchi-topsis based hog parameter selection for complex background sign language recognition. Journal of Visual Communication and Image Representation **71**, 102834 (2020)
15. Mediapipe Solutions guide, https://ai.google.dev/edge/mediapipe/solutions/guide
16. Rupasinghe, R., Ailapperuma, D., De Silva, P., Siriwardana, A., Sudantha, B.: A portable tool for deaf and hearing impaired people. In: ITRU Res. Symp. vol. 6, pp. 25–28 (2014)
17. Taskiran, M., Killioglu, M., Kahraman, N.: A real-time system for recognition of american sign language by using deep learning. In: 2018 41st international conference on telecommunications and signal processing (TSP). pp. 1–5. IEEE (2018)