Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 13, No. 3, September 2025, pp. 668~678
ISSN: 2089-3272, DOI: 10.52549/ijeei.v13i3.5780

Supporting Communication for Deaf People with Sign Language Recognition Using Deep Learning Approach

Thien Ho, Quyen Tran, Tra Nguyen, Nha Tran, Huy Tran
[1]Faculty of Information Technology, Ho Chi Minh City University of Education, Vietnam

Article Info

Article history:

Received Jul 28, 2024
Revised Aug 24, 2025
Accepted Sep 12, 2025

Keywords:

Sign language recognition
Deep learning
Long Short-Term Memory (LSTM)
Multi-layer Perceptron (MLP)

ABSTRACT

Sign language recognition (SLR) plays a crucial role in improving communication for deaf individuals. This paper investigates the recognition of sign language through deep learning models based on action features using Skeleton data from the Argentinian Sign Language (LSA64) dataset. The models explored include Multi-layer Perceptron (MLP) Neural Network, and Long Short-Term Memory (LSTM). The MLP Neural Network, utilizing multiple layers of perceptrons, reached an accuracy of 96.10%. The LSTM model, excelling in processing sequential data, attained the highest accuracy at 98.60%. These results demonstrate the effectiveness of deep learning models in sign language recognition, with LSTM showing the most promise due to its ability to effectively capture temporal dynamics. Consequently, this study opens up prospects for applying sign language recognition technology in practice, contributing to enhancing the quality of life for deaf individuals.

Corresponding Author:

Tran Quang Huy,
Faculty of Information Technology,
Ho Chi Minh City University of Education,
280 An Duong Vuong street, Ward 4, District 5, Ho Chi Minh, Vietnam
Email: huytq@hcmue.edu.vn

1. INTRODUCTION

Sign language is one of the most popular languages that helps the deaf in communication. It is a combination of hand and arm gesture sequences, body posture, and facial expressions, which helps the deaf communicate effectively [1]. However, sign language is the specific language that belongs to deaf community, it is not popular with normal people, which leads to lots of difficulties for deaf individuals to communicate with others. Researching on sign language recognition systems promises to increase significantly in the future based on the number of publications and growing interest in the field [3].

Sign language recognition system also faces many difficulties. One of them is choosing features for sign language recognition models. Compared to action recognition, sign language recognition needs to pay more attention to finger actions and facial expressions. And most of the actions for sign language are performed from the waist up, very few actions are performed from the waist down [4], selecting information is necessary to make the model lighter. Moreover, the suitable dataset for the SLR to train the model is also very limited and sometimes far from reality [5-6]. One of the other difficulties is the differences between sign languages in different countries, requiring a multilingual system capable of translating different sign languages into text or speech.

SLR can be divided into two main types: isolated SLR and continuous SLR. Isolated SLR classifies input into separate signs (sign classfication), while continuous SLR (can be called sign language translation) focuses on the task of translating consecutive sign language. Figure 1 illustrates how isolated and continuous SLR works. When wanting to understand the sentence "I work hard" from the deaf, isolated SLR recognizes and gives prediction results of each sign from the deaf, while continuous SLR continuously recognizes signs from the deaf and produces the result as a complete sentence.

In this paper, we focus on building isolated SLR with the following contributions:

- Extract frame sequences and skeleton data for machine learning models.

- Proposing deep learning models such as ResNet, MLP Neural Network, LSTM to evaluate datasets based on RGB image features and skeleton features.

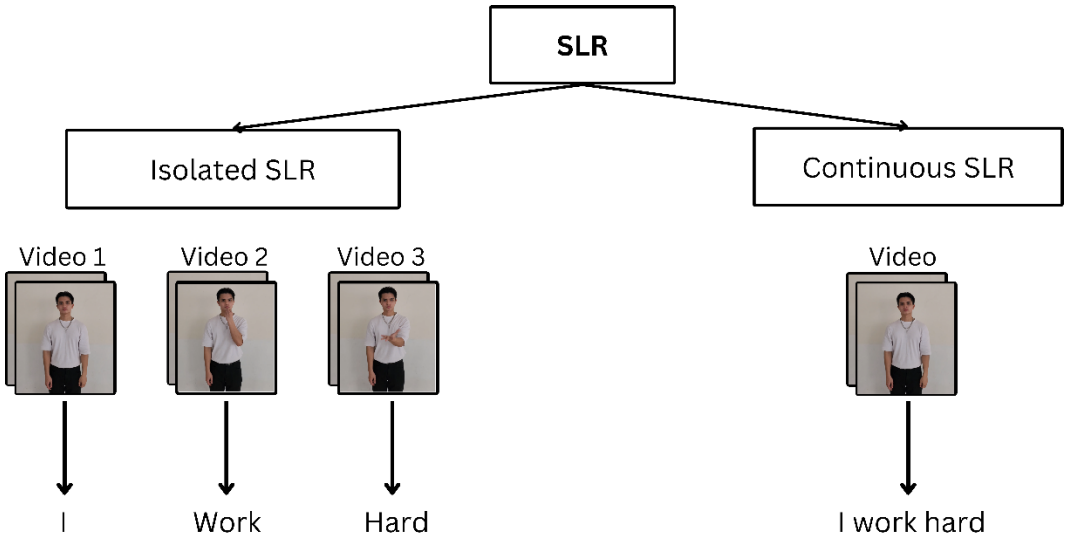

Figure 1. The example compares the identification of isolated and continuous SLR

The remaining content of the paper is organized as follows: some related studies are presented in Section 2; Details of the model building method will be presented in Section 3; Section 4 presents the experiments and results obtained; and the conclusion is given in Section 5.

2. RELATED WORK

In this section, we present an overview of word-level dataset from several countries and some computer vision-based methods for sign language recognition.

2.1. Word-level Sign Language Dataset

There are many different word-level datasets that have been built for sign language recognition in different countries. Table 1 lists the datasets in which videos capture word-level sign language representations. The table includes relevant information such as: number of signs, number of videos, type of video, number of people performing sign language and the name of the country using that sign language.

LSA64 is an Argentinian sign language dataset, with arrangements in context and the clothing worn by the signers. To simplify the problem of dividing the hands in the image, each signer wears fluorescent-colored gloves. This may not be suitable for recognizing in real life. INCLUDE is a dataset about Indian sign language, including 263 signs divided into 15 topics such as places, occupations, animals, etc. The INCLUDE dataset was shot with natural light, without effort to regulate the signer's clothes or signing style. DEVISIGN is a large-scale word-level Chinese sign language dataset, consisting of 2,000 signs and 24,000 samples taken by 8 people in a controlled laboratory environment. Although the number of labels is large (2000 labels), each label only has about 12 samples. Talking about the American Sign Language dataset, the ASLLVD dataset is a well-researched dataset on American Sign Language. It has 2742 signs with distinct meanings performed from 1 to 6 people. However, most labels only have 3 samples, which may affect the model's training. All videos of this dataset are taken with a uniform background for easy hand and face segmentation.

Overall, the above datasets all present different attempts to tackle the task of sign language recognition at the word-level. However, there are also many difficulties when using the dataset due to insufficient number of samples or inappropriateness in practice.

Table 1. Overview of word-level datasets of sign languages

Dataset	Signs	Videos	Type	Signers	Sign Language
LSA64 [1]	64	3200	Video RGB	10	Argentinan
INCLUDE [2]	263	4287	Video RGB	7	Indian
DEVISIGN[3]	2000	24000	Video RGB, depth	8	Chinese
ASLLVD [4]	2742	9794	Video RGB	6	American

2.2. Sign language recognition methods

SLR has made significant progress in recent years. With the emergence of deep learning model-based architectures and advances in computer computing capabilities, it has become possible to design and deploy deep learning models using multimodal data. First, it is essential to choose an appropriate input, which could include video data, skeletal data, or a combination of both. This input data serves as the foundation for the entire recognition process. Second, the system must extract spatial and temporal features from the input data. This involves analyzing the visual and motion characteristics of the sign language gestures, capturing both the shape and movement over time. Finally, the system makes predictions based on the extracted features. All these steps are researched and approached in many directions to increase model's accuracy. Depending on the approach, it is possible for the model to learn different features and combine different features for the model. Inputs for the model can be static or dynamic, RGB, depth, skeleton, flow information [5].

2.2.1. Feature extraction

In previous techniques to extract features for sign language recognition, hand-crafted features are widely used in many studies, for example features based on Histogram of Gradients (HOG) [11-12] and Scale Invariant Feature Transform (SIFT) [13-14]. HOG is a popular method for extracting features from images for use in image recognition and classification methods, it divides the image of the gradient in each cell then creates a vector histogram in each cell. Finally, the histogram vectors of all cells combine to form a characteristic representation of the image. SIFT is an important method in image processing and computer vision to extract and describe features from images in a scale-independent way. SIFT works by identifying keypoints in an image, then describing them based on the gradient directions of pixels around the keypoints. This method provides a robust and scale-invariant feature representation for images, suitable for recognition, image matching, and positioning applications. Besides manual methods, new methods have emerged to extract various features to increase model performance: Depth-based methods use depth images to capture the three-dimensional structure of sign language gestures, enhancing recognition accuracy. These techniques leverage depth variations in hand movements and shapes, as well as depth data from sensors like Kinect, to train neural networks and convolutional neural networks (CNNs) for improved performance in real-time recognition tasks. [15-18], Skeleton-based methods extract and analyze the skeletal structure of the hands and arms to understand and recognize sign language gestures. By capturing the pose and movement of the signer, these techniques utilize models such as Transformers and Long Short-Term Memory (LSTM) networks to achieve high recognition accuracy. Combining skeletal data extraction tools like MediaPipe with machine learning models further enhances the ability to accurately recognize sign language gestures. [19-20], etc.

2.2.2. Sign language recognition method

Previously, there were traditional machine learning approaches for the sign language recognition problem. One of the models that can be mentioned are Hidden Markov Models (HMM). HMM is a stochastic state machine that analyzes time-varying data with spatial and temporal variability [21]. It achieves certain results in sign language recognition [22-25]. However, this method lacks generalizability. With the development of deep learning networks, networks such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short-term Memory (LSTM), Transformer are gradually gaining great attention and are being applied more and more for SLR. While CNN is capable of extracting information from images, models such as RNN, LSTM, and Transformer perform more strongly in sequence information. CNN can be used by research as a feature extraction method of video frames or as a classifier [18, 23-25]. Models capable of learning good sequential information from videos such as LSTM, Transformers. B. Sundar et al. [20] proposed a user recognition model using LSTM and MediaPipe to recognize 26 English letters with F1-Score reaching 99%, the proposed model has the potential to be highly effective when used in human-computer interaction (HCI). M. Bohacek et al. [19] proposed the SPOTER model based on the Transformer architecture. The research team realized the potential of the Transformer when it comes to relatively low computational costs and outstanding performance in string processing tasks. This makes them the perfect choice for a lightweight computing solution capable of running on modern mobile devices. This model has strong body position normalization and enhancements compared to previous models, significantly improving accuracy. SPOTER was tested and compared on 2 datasets. LSA64, WLASL data. Recent works also use 3D convolutional networks to be able to learn both spatial and temporal features [26-27].

3. DATASET

The LSA64 dataset is an extensive and meticulously curated collection aimed at advancing the field of Argentinian Sign Language recognition. This dataset offers a diverse range of contextual arrangements and includes signers with varied clothing to enhance the complexity and applicability of the data. Notably, each signer wears fluorescent-colored gloves, a deliberate choice to simplify the segmentation of hands from the

background in images. While this feature aids in the technical processing of the data, it might not fully represent real-world conditions where such gloves are not worn.

The primary objective behind the creation of the LSA64 dataset is twofold: to develop a comprehensive dictionary for Argentinian Sign Language (LSA) and to train robust automatic sign recognizers. The dataset contains 3200 video recordings, featuring 10 non-expert subjects performing 5 repetitions of 64 distinct signs. Each class label includes 50 videos. The 64 classes in the dataset represent commonly used signs in everyday communication which are detailed in Table 2. The signs included in the LSA64 dataset were selected based on their prevalence in everyday communication, ensuring the dataset's practical applicability.

Table 2. Labels of the LSA64 dataset

Name	Hand	Name	Hand	Name	Hand	Name	Hand
Opaque	Right	Call	Right	Hungry	Right	Yogurt	Both
Red	Right	Skimmer	Right	Map	Both	Accept	Both
Green	Right	Bitter	Right	Coin	Both	Thanks	Both
Yellow	Right	Sweet milk	Right	Music	Both	Shut down	Right
Bright	Right	Milk	Right	Ship	Right	Appear	Both
Light-blue	Right	Water	Right	None	Right	To land	Both
Colors	Right	Food	Right	Name	Right	Catch	Both
Pink	Right	Argentina	Right	Patience	Right	Help	Both
Women	Right	Uruguay	Right	Perfume	Right	Dance	Both
Enemy	Right	Country	Right	Deaf	Right	Bathe	Both
Son	Right	Last name	Right	Trap	Both	Buy	Right
Man	Right	Where	Right	Rice	Both	Copy	Both
Away	Right	Mock	Both	Barbecue	Both	Run	Both
Drawer	Right	Birthday	Right	Candy	Right	Realize	Right
Born	Right	Breakfast	Both	Chewing-gum	Right	Give	Both
Learn	Right	Photo	Both	Spaghetti	Both	Find	Right

Each video captures a signer performing a sign in a controlled environment, ensuring high-quality and consistent visual data. This controlled setting considers various factors such as lighting conditions and signer variability, which are crucial for developing and evaluating machine learning models dedicated to sign language recognition. The inclusion of multiple repetitions by different signers introduces variability that is essential for training models capable of generalizing across different individuals and conditions.

The LSA64 dataset stands as a vital resource for researchers and developers in the domains of computer vision and natural language processing, particularly those focusing on sign language recognition and translation. Its comprehensive and detailed nature makes it suitable for various applications, including the development of real-time sign language recognition systems, enhancing communication accessibility for the deaf and hard-of-hearing community, and contributing to the creation of more inclusive human-computer interaction technologies.

By offering a well-rounded set of sign language videos, the LSA64 dataset not only supports the development of advanced machine learning algorithms but also paves the way for significant improvements in the accessibility and usability of sign language technologies in everyday life.

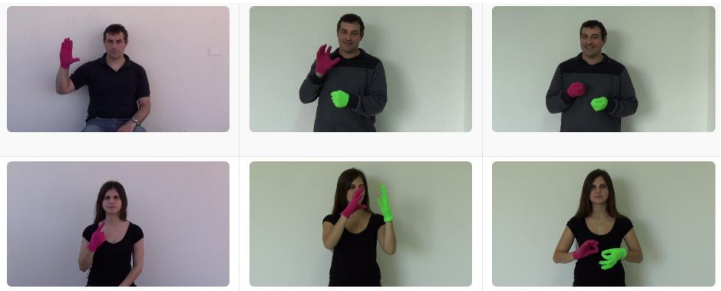

Figure 2. Snapshot of six different sign of LSA64 dataset.

4. METHOD

In this section, we build a model to recognize sign language for deaf people divided into two main directions. Skeleton-based sign language recognition and spatial-based sign language recognition. To recognize sign language based on the skeleton, we proceed to extract the skeleton using MediapPipe. We then proceed to extract skeleton features. To evaluate the skeleton recognition model, we use two models: MLP Neural Network and LSTM. To recognize sign language based on spatial feature, we extract frames from the videos, then extract them into spatial features and use the ResNet model to evaluate.

4.1. Skeleton Extraction

We use the MediaPipe Holistic library [29] from MediaPipe to extract data about body postures from videos. MediaPipe is a library built to optimize performance on many devices, including devices with limited resources such as mobile phones, providing real-time performance with high processing speeds and low resource consumption.

We use the MeadiaPipe Hand model to extract points on the hand. The keypoint extracted from MediaPipe are shown in the Figure 3. We choose keypoints numbered 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20.

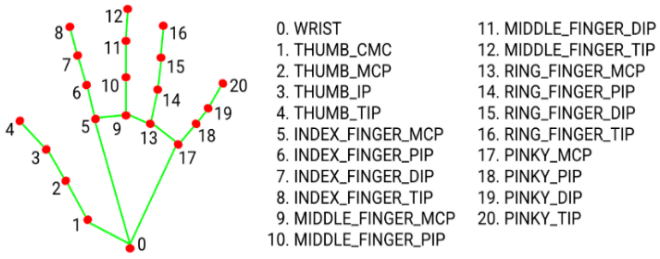

Figure 3. Hand landmarks are extracted from MediaPipe [29]

Each keypoint is represented by a pair of coordinates (x, y) corresponding to the location on the image. All keypoints are organized into a vector of size 42, where each pair of coordinates (x, y) contributes to the representation of a particular pose of a frame. In case there is no performer in the frame or MediaPipe cannot identify some keypoints, we decided to fill the unknown coordinates with a value of 0.

To minimize overfitting and enhance the learning ability of the model, we apply the random rotation technique introduced [19] in the processing of training data. Eq. (1) shows the calculation formula for this technique.

The process of random rotation of coordinates is performed from 0° to 13° and is applied according to a specific formula. In this way, we will optimize the uniformity and flexibility of the training data, helping the model learn from many different angles and poses, improving generalization ability when applying the model in practice.

$$f_{rotate}(x,y) = ((x-0.5)cos\theta - (y-0.5)sin\theta + 0.5, \\ (y-0.5)cos\theta + (x-0.5)sin\theta + 0.5) \quad (1)$$

4.2. MLP Neural Network

The model consists of a sequence of neural network layers, starting with a Dense layer with 256 units and an activation function of relu. This layer converts the input into a feature space of decreasing dimensionality. We then add a BatchNormalization layer to normalize the output of the previous layer, which improves learning speed and model stability. Next, we add another Dense layer with 128 units and an activation function of relu. This layer further transforms the feature space, helping the model learn more complex features. We then add another BatchNormalization layer to further normalize the output of the previous layer, increasing the stability of the model. We continue by adding a Dropout layer with a dropout rate of 0.5, which helps prevent overfitting by randomly dropping half the units during training. Then we add two more Dense layers with 64 units and an activation function of relu. Both layers continue to transform the feature space in preparation for the final layer. Finally, we add a final Dense layer with 64 output units and a softmax activation function. This layer converts the output into a probability distribution over 64 output classes, representing 64 different classes. Figure 4 show MLP model architecture.

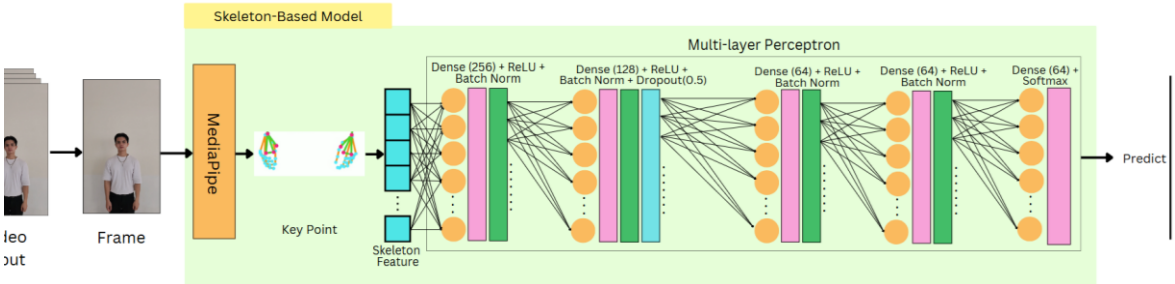

Figure 4. MLP with skeleton features.

4.3. LSTM model

We start by adding an LSTM layer with 512 input data units that are a variable length string and each sample has 18 features. Then apply the Batch Normalization layer to normalize and optimize the previous layer. This helps improve the convergence speed and stability of the model during training. Next, add another LSTM layer with 256 units. Then add a Dropout layer with a dropout rate of 0.5 to prevent overlearning of the model and reduce overfitting. We continue by adding another LSTM layer with 128 units and continue adding another Dropout layer with a rejection ratio of 0.5. Finally, we add a final LSTM layer with 64 units and continue with a BatchNormalization layer to normalize the output of the previous layer. Finally, add a Dense (fully connected) layer with 64 output units and a softmax activation function. This layer converts the output from a number into a probability distribution over 64 output classes, representing 64 different classification classes. Figure 5 show LSTM model architecture.

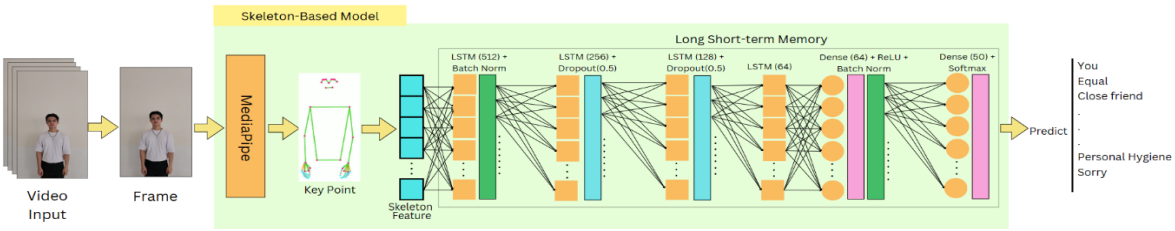

Figure 5. LSTM with skeleton features.

4.4. ResNet50 Model

Deep neural networks often have a large number of parameters. Therefore, the dataset needs to be large enough to train the model parameters and prevent overfitting. For small datasets, the parameters will overfit the data. Pre-training can be a kind of regularization, minimizing variance, avoiding overfitting on small datasets [30].

The ResNet network includes two main blocks: the "Basic Block" and the "Bottleneck Block." The Basic Block uses 3x3 convolutional layers, while the Bottleneck Block uses smaller convolutional layers to reduce network complexity. Each block in ResNet consists of multiple convolutional layers stacked together, combined with Batch Normalization layers and ReLU activation functions. "Skip connections" are added to each block to form residual connections, allowing information to be transmitted directly across the network. This minimizes information loss and increases the model's robustness. The input image is first fed into the network through the initial convolutional layers, then normalized with Batch Normalization and activated with the ReLU function. Subsequently, the blocks of the ResNet network are executed sequentially, each containing a series of convolutional layers along with Batch Normalization and ReLU layers. During the forward pass, residual connections facilitate the easy passage of information through the blocks, enabling the network to learn more complex data representations. Finally, the network output, after traversing the blocks, is passed into a Fully Connected Layer to classify the image into predefined classes. Figure 7 illustrates the ResNet model architecture.

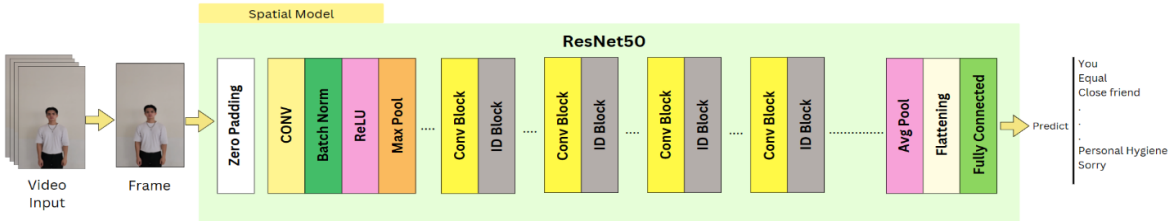

Figure 6. ResNet50 with spatial features.

5. RESULTS AND DISCUSSION

We conduct experiments on the Google Colab Pro environment, utilizing a GPU (T4 with 15GB VRAM), 12.7GB of RAM, and 201.2GB of disk space. The experiments are performed across three models: ResNet50, MLP, and LSTM. To comprehensively evaluate the performance of these models, we employ several metrics, including accuracy (2), precision (3), recall (4) and F1-scores (5). These metrics provide a well-rounded assessment of the models' effectiveness in recognizing and classifying Argentinian Sign Language gestures. Accuracy measures the overall correctness of the model, precision indicates the proportion

of true positive predictions among all positive predictions, recall assesses the model's ability to identify all relevant instances, and the F1 score balances precision and recall, offering a single metric that reflects both. This thorough evaluation ensures a robust analysis of each model's performance in our experiments.

$$\text{Accuracy (acc)} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \quad (2)$$

$$\text{Precision (P)} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (3)$$

$$\text{Recall (R)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4)$$

$$\text{F1} - \text{score} = \frac{2 \text{ x } (\text{P x R})}{(\text{P} + \text{R})} \quad (5)$$

TP (True Positive): The count of correctly predicted positive cases.

TN (True Negative): The count of correctly predicted negative cases.

FP (False Positive): The count of instances where observations labeled as negative are incorrectly predicted as positive.

FN (False Negative): The count of instances where observations labeled as positive are incorrectly predicted as negative.

With ResNet50 from Keras library with parameters epochs, batch size, learning rate are shown in Table 3.

The MLP and LSTM have epochs, batch size, and learning rate parameters shown in Table 4, Table 5.

Table 3. Training parameters for ResNet

Hyperparameter	Value
epochs	500
batch-size	25
learning rate	0.0001

Table 4. Training parameters for MLP

Hyperparameter	Value
epochs	60
batch-size	32
learning rate	Adam (0.0001)

Table 5. Training parameters for LSTM

Hyperparameter	Value
epochs	60
batch-size	32
learning rate	Adam (0.0001)

The video dataset, approximately 3.7 Gigabytes (GB) in size and containing 3200 videos, has been divided into two parts: a training set and a test set for both skeleton-based and spatial feature-based models. More information is available in Table 6.

Table 6. Rates of train and test

	Train set	Valid set	Test set
Rate	70%	10%	20%
Videos	2240	320	640
Total video		3200	

The results show that the LSTM model outperformed the others, achieving an accuracy of 98.60%, demonstrating its superior ability to capture the temporal dynamics inherent in sign language. The MLP model also showed strong performance with an accuracy of 96.10%. In contrast, the ResNet50 model achieved an accuracy of 82.13%, suggesting that while ResNet50 is effective in handling spatial features, it may not be as efficient in recognizing the sequential nature of sign language compared to models specifically designed for temporal data, such as LSTM. Detailed performance metrics for these models are presented in Table 7, which provides a comprehensive comparison of the accuracy rates of MLP, and LSTM on the LSA64 dataset.

Our experimental results show that our model achieves very high accuracy across most labels, with many labels reaching perfect accuracy of 100%. Labels such as Red, Light_blue, Colors, Women, Away, Drawer, Learn, Milk, Food, Uruguay, Country, Where, Mock, Birthday, Photo, Music, Ship, Perfume, Barbecue, Candy, Yogurt, Bathe, Buy, Copy, Realize, Give and Help all achieved perfect accuracy of 100%.

This is an encouraging result, indicating that the model can classify these labels accurately without any difficulty. Other labels like Green, Yellow, Bright, Enemy, Born, Call, Bitter, Sweet_milk, Water, Last_name, Map, Coin, Trap, Rice, Chewing_gum, Spaghetti, Accept, Shut_down, Appear, To_land, Find and Run also achieved very high accuracy, ranging from 97% to 99%, indicating high reliability.

However, there are still some labels with accuracy below 95%, such as Opaque 93%, Skimmer 92%, Hungry 93% and Shut_down 93% as shown in Table 9. These labels might need further improvement to achieve higher accuracy. The reasons could be due to insufficient diversity in the training data for these labels or an inadequate number of samples for these labels. Focusing on these labels during additional training could help improve the overall accuracy of the model.

Table 8 shows the comparison between different models on the LSA64 dataset. The SPOTER model achieved the highest accuracy at 100%, followed by the HMM model at 97.44% and the 3DGCN model at 94.84%. Although our LSTM model did not achieve the highest accuracy compared to SPOTER, it still outperformed many other methods, highlighting its strong potential.

Table 7. Accuracy of the models

Network	Accuracy (%)	Precision	Recall	F1-score
Skeleton MLP	96.10	95.92	95.84	95.81
Skeleton LSTM	98.60	98.45	98.46	98.44
RGB Resnet	82.13	82.15	82.10	82.12

Table 8. Comparison with other model on LSA64 data

Method	nClasses	Accuracy (%)
SPOTER [19]	64	100
3DGCN	64	94.84
HMM [32]	64	97.44
Skeleton MLP (Ours)	64	96.10
Skeleton LSTM (Ours)	64	98.60
RGB Resnet (Ours)	64	82.13

Table 9. Results of each label on the LSTM model

Label	Accuracy	Label	Accuracy
Opaque	93.00	Milk	100.00
Red	100.00	Water	99.00
Green	98.00	Food	100.00
Yellow	99.00	Argentina	95.00
Bright	95.00	Uruguay	99.00
Light_blue	100.00	Country	100.00
Colors	100.00	Last_name	99.00
Pink	95.00	Where	100.00
Women	100.00	Mock	100.00
Enemy	99.00	Birthday	100.00
Son	95.00	Breakfast	98.00
Man	97.00	Photo	100.00
Away	100.00	Hungry	93.00
Drawer	100.00	Map	98.00
Born	98.00	Coin	99.00
Learn	100.00	Music	96.00
Call	99.00	Ship	99.00
Skimmer	92.00	none	99.00
Bitter	98.00	Name	99.00
Sweet_milk	99.00	Patience	100.00
Perfume	100.00	Appear	99.00
Deaf	97.00	To_land	98.00
Trap	99.00	Catch	100.00
Rice	99.00	Help	99.00
Barbecue	100.00	Dance	99.00
Candy	100.00	Bathe	100.00
Chewing_gum	99.00	Buy	100.00
Spaghetti	97.00	Copy	100.00
Yogurt	100.00	Run	99.00
Accept	98.00	Realize	100.00
Thanks	100.00	Give	100.00
Shut_down	93.00	Find	98.00

Our findings reveal that models based on skeletal features not only show shorter training times such as MLP with 30 min 17 seconds, LSTM with 45 min 18 seconds but also achieve better accuracy. higher body. The LSTM model, with its ability to achieve accuracy up to 98.60%, demonstrates superiority in capturing and explaining the sequential and dynamic aspects of sign language. In contrast, the spatial feature-based model (ResNet50) required a longer training time of 2 hours 15 minutes 10 seconds and yielded slightly lower accuracy (82.13%). In summary, our model has shown very high and stable performance across most labels, with many achieving perfect accuracy. However, there are still areas for improvement to enhance the accuracy of the lower-performing labels. These results demonstrate the potential of our model in accurately classifying a diverse and rich dataset, emphasizing the importance of choosing appropriate features for different types of models.

6. CONCLUSION

This study investigates the enhancement of sign language recognition (SLR) systems for the deaf community through the application of deep learning models. We evaluated three models ResNet50, MLP, and LSTM using the LSA64 dataset. The LSTM model outperformed the others, achieving an accuracy of 98.60%, thereby showcasing its superior capability in capturing the temporal dynamics inherent in sign language. The MLP and ResNet50 models also exhibited robust performance, with accuracies of 96.10% and 82.13%, respectively. Our findings indicate that deep learning models, particularly those leveraging skeletal features, hold significant potential for advancing SLR systems. Future research should prioritize the expansion of datasets and the optimization of these models for real-world applications, ultimately aiming to improve communication accessibility for the deaf community.

REFERENCES

[1] H. Cooper, B. Holt, and R. Bowden, "Sign Language Recognition," in Visual Analysis of Humans: Looking at People, T. B. Moeslund, A. Hilton, V. Krüger, and L. Sigal, Eds., London: Springer, 2011, pp. 539–562.

[2] GSO, "The National Survey on People with Disabilities 2016 (VDS2016)," Ha Noi, Viet Nam: General Statistics Office., Final Report.

[3] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," Arch Computat Methods Eng, vol. 28, no. 3, pp. 785–813, May 2021.

[4] Ursula Bellugi, U. Bellugi, Susan D. Fischer, and S. D. Fischer, "A comparison of sign language and spoken language," Cognition, vol. 1, no. 2, pp. 173–200, Jan. 1972.

[5] Razieh Rastgoo, R. Rastgoo, Kourosh Kiani, K. Kiani, Sérgio Escalera, and S. Escalera, "Sign Language Recognition - A Deep Survey.," Expert Systems With Applications, vol. 164, p. 113794, Feb. 2021.

[6] A. Núñez-Marcos, Adrián Núñez-Marcos, Olatz Perez-de-Viñaspre, Olatz Perez-de-Viñaspre, G. Labaka, and Gorka Labaka, "A survey on Sign Language machine translation," Expert Systems With Applications, vol. 213, pp. 118993–118993, Oct. 2022, doi: 10.1016/j.eswa.2022.118993.

[7] F. Ronchetti, F. Quiroga, C. A. Estrebou, L. C. Lanzarini, and A. Rosete, "LSA64: An Argentinian Sign Language Dataset," arXiv.org, Jan. 2016.

[8] A. Sridhar, R. G. Ganesan, P. Kumar, and M. Khapra, "INCLUDE: A Large Scale Dataset for Indian Sign Language Recognition," in Proceedings of the 28th ACM International Conference on Multimedia, in MM '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1366–1375.

[9] H. Wang, X. Chai, X. Hong, G. Zhao, and X. Chen, "Isolated Sign Language Recognition with Grassmann Covariance Matrices," ACM Trans. Access. Comput., vol. 8, no. 4, p. 14:1-14:21, May 2016.

[10] V. Athitsos et al., "The American Sign Language Lexicon Video Dataset," in 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Jun. 2008, pp. 1–8.

[11] H. Wang, X. Chai, Y. Zhou, and X. Chen, "Fast sign language recognition benefited from low rank approximation," in 2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), May 2015, pp. 1–6.

[12] J. He, Z. Liu, and J. Zhang, "Chinese sign language recognition based on trajectory and hand shape features," in 2016 Visual Communications and Image Processing (VCIP), Nov. 2016, pp. 1–4.

[13] N. H. Dardas and N. D. Georganas, "Real-Time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques," IEEE Transactions on Instrumentation and Measurement, vol. 60, no. 11, pp. 3592–3607, Nov. 2011.

[14] A. Tharwat, T. Gaber, A. E. Hassanien, M. K. Shahin, and B. Refaat, "SIFT-Based Arabic Sign Language Recognition System," in Afro-European Conference for Industrial Advancement, A. Abraham, P. Krömer, and V. Snasel, Eds., Cham: Springer International Publishing, 2015, pp. 359–370.

[15] L. Zheng and B. Liang, "Sign language recognition using depth images," in 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), Nov. 2016, pp. 1–6.

[16] V. Tiwari, V. Anand, A. G. Keskar, and V. R. Satpute, "Sign language recognition through kinect based depth images and neural network," in 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Aug. 2015, pp. 194–198.

[17] B. Kang, S. Tripathi, and T. Q. Nguyen, "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map," in 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Nov. 2015, pp. 136–140.
[18] T. Raghuveera, R. Deepthi, R. Mangalashri, and R. Akshaya, "A depth-based Indian Sign Language recognition using Microsoft Kinect," Sādhanā, vol. 45, no. 1, p. 34, Jan. 2020.
[19] Matyáš Boháček and Marek Hrúz, "Sign Pose-based Transformer for Word-level Sign Language Recognition," 2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW), Jan. 2022.
[20] B. Shyam Sundar and T. Bagyammal, "American Sign Language Recognition for Alphabets Using MediaPipe and LSTM," Procedia Computer Science, vol. 215, pp. 642–651, Jan. 2022.
[21] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," Pattern Recognition, vol. 36, no. 3, pp. 585–601, Mar. 2003.
[22] N. A. Sarhan, Y. El-Sonbaty, and S. M. Youssef, "HMM-based Arabic Sign Language Recognition using Kinect," in 2015 Tenth International Conference on Digital Information Management (ICDIM), Oct. 2015, pp. 169–174.
[23] Anil Osman Tur, A. O. Tur, Hacer Yalım Keleş, and H. Y. Keles, "Evaluation Of Hidden Markov Models Using Deep CNN Features In Isolated Sign Recognition," Multimedia tools and applications, Jun. 2020.
[24] T. Starner, J. Weaver, and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 12, pp. 1371–1375, Dec. 1998.
[25] J. Zhang, W. Zhou, C. Xie, J. Pu, and H. Li, "Chinese sign language recognition with adaptive HMM," in 2016 IEEE International Conference on Multimedia and Expo (ICME), Jul. 2016, pp. 1–6.
[26] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "C3D: Generic Features for Video Analysis," ArXiv, Dec. 2014.
[27] P. M. Ferreira, J. S. Cardoso, and A. Rebelo, "Multimodal Learning for Sign Language Recognition," in Pattern Recognition and Image Analysis, L. A. Alexandre, J. Salvador Sánchez, and J. M. F. Rodrigues, Eds., Cham: Springer International Publishing, 2017, pp. 313–321.
[28] "Thông tư ban hành quy định chuẩn về ngôn ngữ ký hiệu cho người khuyết tật." Hà Nội, Việt Nam.
[29] Google, "MediaPipe Holistic," GitHub. Accessed: Mar. 29, 2024. [Online]. Available: https://github.com/google/mediapipe/blob/master/docs/solutions/holistic.md
[30] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why Does Unsupervised Pre-training Help Deep Learning?," in Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, Mar. 2010, pp. 201–208.
[31] M. Al-Hammadi *et al.*, "Spatial Attention-Based 3D Graph Convolutional Neural Network for Sign Language Recognition," *Sensors*, vol. 22, no. 12, Art. no. 12, Jan. 2022, doi: 10.3390/s22124558.
[32] F. Ronchetti, F. Quiroga, C. Estrebou, L. Lanzarini, and A. Rosete, "Sign Languague Recognition Without Frame-Sequencing Constraints: A Proof of Concept on the Argentinian Sign Language," in *Advances in Artificial Intelligence - IBERAMIA 2016*, M. Montes y Gómez, H. J. Escalante, A. Segura, and J. de D. Murillo, Eds., Cham: Springer International Publishing, 2016, pp. 338–349. doi: 10.1007/978-3-319-47955-2_28.

BIOGRAPHY OF AUTHORS

Thien Ho He is currently a 3rd-year student at the Ho Chi Minh City University of Education, Ho Chi Minh City, Vietnam. His research includes machine learning, deep learning, and computer vision. He can be contacted at email: hosithien13723@gmail.com

Quyen Tran She is currently a 3rd-year student at the Ho Chi Minh City University of Education, Ho Chi Minh City, Vietnam. Her research includes machine learning, deep learning, and computer vision. She can be contacted at email: trantuquyen19@gmail.com

Tra Nguyen She is currently a 2rd-year student at the Ho Chi Minh City University of Education, Ho Chi Minh City, Vietnam. Her research includes machine learning, deep learning, and computer vision. She can be contacted at email: nguyentra2k4@gmail.com

Nha Tran received BS. Computer Science (2014), MS. Computer Science (2020) from HCM University of Education. He is currently a lecturer of Information Technology Faculty, Ho Chi Minh city University of Education. He is currently pursuing the Ph.D. degree in the University of Information Technology, Ho Chi Minh city. His research interest include computer vision, information retrieval, affective computing, machine learning. He was recipient of the Best Paper Award from ICAEIC 2020. You can contact him via email: nhatt@hcmue.edu.vn.

Huy Tran MS. Computer Science (2020) from HCM University of Education. He is currently a lecturer of Information Technology Faculty, Ho Chi Minh city University of Education. His research interest include action reconigtion, question answering and deep learning. You can contact him via email: huytq@hcmue.edu.vn.