JST: Smart Systems and Devices
Volume 35, Issue 1, January 2025, 010-019

Vietnamese Sign Language Alphabet Recognition Using Deep Learning and Mediapipe Methods

Tran Anh Vu[1], Phung Van Kien[1], Hoang Quang Huy[1*], Pham Thi Viet Huong[2]
[1]Hanoi University of Science and Technology, Ha Noi, Vietnam
[2]International School, Vietnam National Unviersity, Ha Noi, Vietnam
[*]Corresponding author email: huy.hoangquang@hust.edu.vn

Abstract

Sign language serves as a vital communication method for individuals with hearing impairments, relying on hand movements and gestures to convey meaning. For centuries, it has enabled interaction for people with hearing and speech disabilities. However, despite its historical significance, many individuals in society struggle to interpret these signs, creating a communication barrier with the deaf and mute community. This paper proposes a deep learning-based system specifically designed to recognize Vietnamese Sign Language (VSL) gestures. The dataset developed includes 23 alphabet signs and 2 accent marks unique to VSL, with 22 of the alphabet signs resembling those in English. The proposed system achieves an accuracy exceeding 91% on the raw dataset and 95% on the processed dataset.

Keywords: Vietnamese Sign Language, Mediapipe, keypoint, image processing, deep learning.

1. Introduction

Sign language, a visual mode of communication, comprises hand signals, gestures, facial expressions, and body language. Evidence suggests that sign language has been a widely used communication method for people with hearing impairments since at least the fourth century BC [1]. However, a large portion of the general population is not familiar with sign language, making it difficult for individuals relying on this form of communication to interact without the help of an interpreter.

Sign language recognition has therefore been a topic of research for many years. Vietnamese Sign Language (VSL), though influenced by American Sign Language (ASL), includes several unique gestures not found in ASL [2]. Moreover, the lack of a publicly available VSL dataset has limited the number of studies dedicated to recognizing it. To address this gap, we constructed a VSL dataset by recording hand gesture videos using mobile phone cameras and applying data processing techniques to extract still images.

Gesture recognition (GR) can be categorized into two types: static gestures and dynamic gestures [3]. Static gestures are represented by single images, while dynamic gestures require a sequence of images for recognition. Feature extraction is a critical phase in pattern recognition, particularly in static gesture recognition, where the visual elements of an image serve as the most reliable indicators for classification [4].

This paper leverages the constructed dataset to develop a deep learning-based system for recognizing VSL Alphabet. The proposed approach achieves an accuracy rate of approximately 92-96%.

2. Relate Works

Different machine learning techniques such as K-nearest neighbours (KNN), Random Forest [5], Support Vector Machine (SVM) [6], have been used to classify the data into alphabets and words. As a result of the recent progress of computers, deep learning-based methods have been extensively investigated in the vision-based approach. A Convolutional Neural Networks (CNN) based hand gesture recognition system was proposed in [4]. To increase the system's robustness, skin modelling, hand positioning, and orientation calibration were added. Dataset photos were registered using the Xbox Kinect camera. The research in [7] further suggested complex neural networks like CNN and stacked denoising auto encoder that are more effective in learning complex hand motions with fewer errors. The 24 hand motions in Moeslund's database were recognized using deep learning. Recognition has been accomplished in [8] using both standard feature extraction approaches and CNN. Sign colour images were processed with an experimental hybrid discrete wavelet transform Gabor filter by the authors. The authors evaluated a variety of classifiers, including the Random Forest, K-Nearest Neighbours, and Support Vector Machine, for their performance. Then, a CNN was applied to get an accuracy of 97.01%.

ISSN 2734-9373
https://doi.org/10.51316/jst.179.ssad.2025.35.1.2
Received: Oct 12, 2024 ; revised: Oct 31, 2024
accepted: Oct 31, 2024

From what was mentioned above, convolution neural networks have proven its effectiveness in classifying hand gestures. After some re-testing experiments, we decide to choose convolution neural networks for our own dataset.

The current trend in gesture recognition (GR) research can be divided into two main approaches. The first involves the use of hand-worn or sensor-based devices. In this method, electronic gloves equipped with sensors are utilized to capture gesture data, which is then analyzed and categorized [9, 10]. While this approach is more robust and accurate, its practicality is limited by the need for specialized equipment. The second approach relies on vision-based techniques. Here, the process begins with capturing gesture postures or images using a camera. These images are subsequently processed using various image processing techniques, such as segmentation, matching, recognition, and classification [11, 12]. Owing to the minimal requirement for specialized hardware, this approach has attracted considerable attention from researchers [7].

In the second category, some GR approaches include different algorithms such as: an edge-oriented histogram-based method [13], combination of the histogram of oriented gradients (HOG) and the scale-invariant feature transform (SIFT) based method [14], deep learning-based methods [15-17], and the scale-invariant feature transform (SIFT) based method [6]. For static gesture identification, an edge-oriented histogram was used in [13]. Sign language alphabets' edge histogram counts were used as features, in addition to that, a multiclass SVM classifier was applied and attained an overall accuracy of 93.75%. Using the HOG and SIFT algorithms, Gupta *et al*. [14] can recognize hand gestures. They combined the HOG and SIFT characteristics for categorization into a single array. The classification process makes use of a typical KNN classifier. For double-handed gestures, 179 of the 200 movements were successfully classified, and 59 of the 60 gestures were correctly classified for single-handed motions.

Traditional feature extraction methods often fail to retain critical information that could enhance object classification. The K-Nearest Neighbors (KNN) algorithm, while straightforward, is inherently memory-intensive and becomes impractical for classifiers operating on large-scale datasets due to its reliance on storing and referencing the entire training data without learning underlying patterns. Current approaches predominantly focus on extracting basic and simplified features, often relying on image transformation algorithms to optimize model performance. Alternatively, some methods prioritize developing proprietary datasets with improved prerocessing strategies to enhance classification accuracy.

In the context of VSL, available datasets remain relatively limited, as shown in Table 1, and the use of bone diagrams has not been widely explored in recent VSL recognition research. A bone diagram is a representation of the hand by connecting keypoints (landmarks) on the hand, similar to how bones are connected at the joints. Therefore, in this paper, we propose a novel approach to address this issue by utilizing bone diagram to recognize signs. Our focus is on the preprocessing phase to ensure clean and reliable data. By leveraging OpenCV and Mediapipe to process raw data, we handle challenging backgrounds and extract hand images into keypoints. Finally, the keypoints are fed into the classification model for recognition.

Table 1. A summary about some datasets

Authors	Dataset	Contents
Hassene Ben Amara [2]	20BN-JESTER	Video sequences of 27 hand gestures represented for pre-defined hand gestures
A.H. Vo, [5]	VSL-WRF	Two dataset contains total 27 Vietnamese words from the topic of relative family topic
O.K. Oyedotun and A. Khashman [10]	PRIMA	24 different static hand gestures of Latin alphabets
S. Nagarajan and T. Subashini [11]	ASL Alphabet	Self-captured of 24 static hand gestures of American sign language

We collected data from 35 trained individuals, totaling approximately 200 minutes across 875 videos, which corresponds to nearly 358,000 image samples. This extensive dataset is expected to enhance the model's ability to accurately recognize image samples.

3. Proposed Method

The proposed method comprises three main stages: (1) Data collecting, (2) Image Rre-processing, and (3) Model Training for classification. The deep learning model is based on a CNN, as detailed in [5]. The workflow is illustrated in the block diagram shown in Fig. 1.

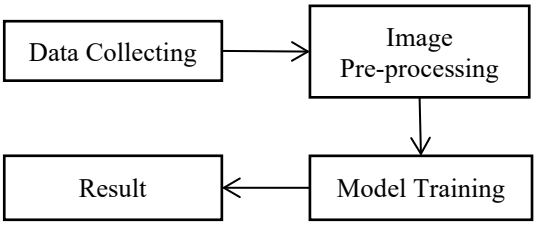

Fig. 1. Block Diagram of the Proposed Study

3.1. Data Collecting

3.1.1. A comparison between ASL and VSL

In this study, we build our own VSL dataset. In order to do that, we consider the differences between VSL and ASL.

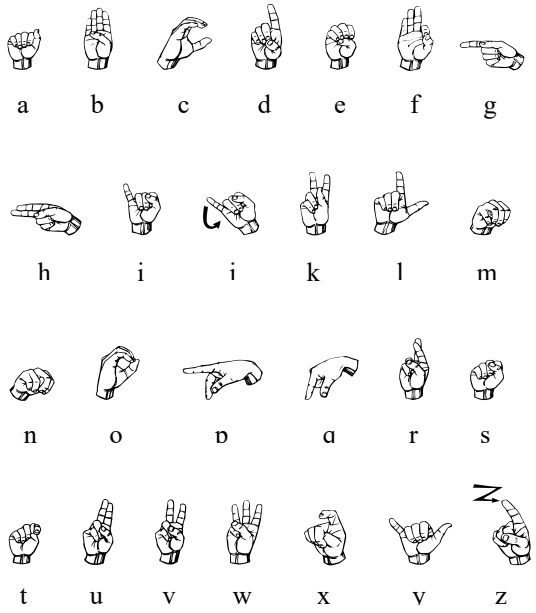

Fig. 2. American Sign Language Alphabet [18]

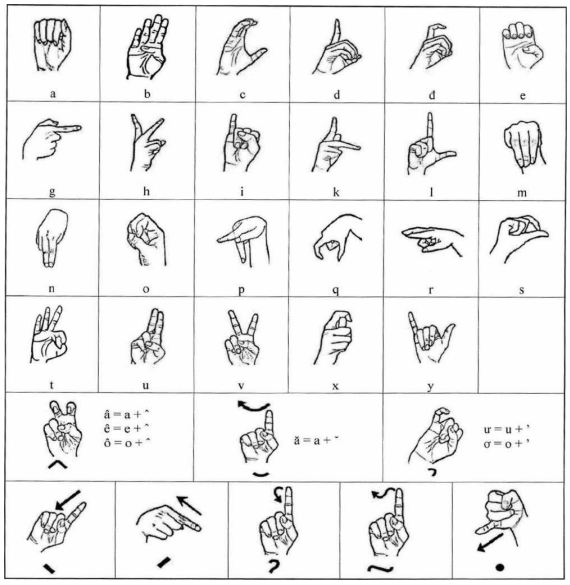

Fig. 3. Vietnamese sign language alphabet [18]

ASL alphabet consists of 26 characters: “A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z”. 24 of them can be described as a static hand pose, except for J and Z need to be described as a dynamic hand motion [18].

Meanwhile, VSL alphabet is quite different to any sign language alphabet. VSL has some additional accent marks and lack of some common characters of Latin alphabet. The standard of VSL was introduced by Vietnamese Ministry of Education and Training [19].

As shown in Fig. 2 and Fig. 3, we can see that according to Vietnamese standard, there are 22 letters of VSL that are similar to ASL, 11 of them have the same hand pose including “A, B, C, G, L, O, P, Q, U, V, Y”. The remaining 11 letters, including “D, E, H, K, M, N, R, S, T, X” have different shapes.

Apart from 22 similar letters, VSL removes 4 letters of ASL (“F, J, W, Z”) and has a unique letter Đ (D with stroke - dyet). Moreover, VSL also has 3 additional accent marks: “ ^ ” (the circumflex), “ ˘ ” (the breve), and “ ’ ” (the horn). Those marks create 11 single vowels in Vietnamese by the formula in the 5th row of Fig. 3. Vietnamese also has 5 tonal symbols in the 6th row of Fig. 3: “ ` ” (deep), “ ´ ” (sharp), “ ̉ ” (asking), “ ~ ” (tumbling) and “ ̣ ” (heavy). The differences between VSL and ASL are summarized in Fig. 4.

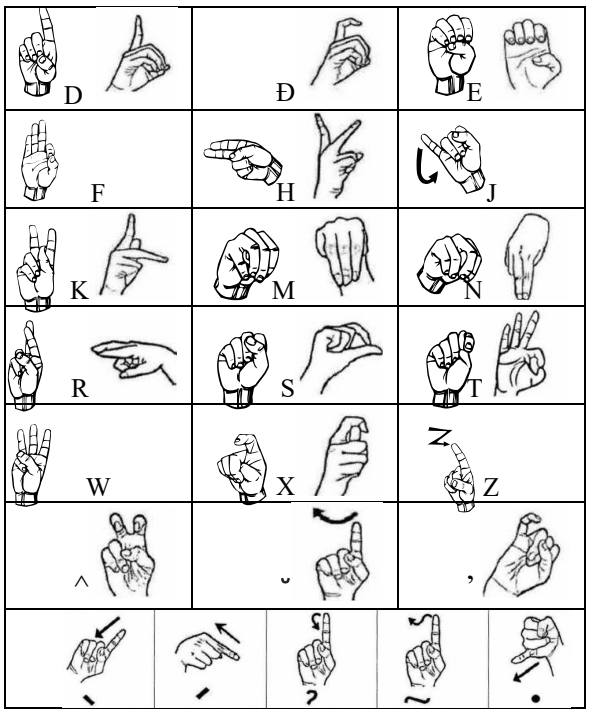

Fig 4 A comparision of differential symbols between ASL (left, 5 first rows) and VSL (right, 5 first rows; 6-7th row VSL only)

In VSL standard, there are 23 letters using static hand pose to describe. Two of additional accent marks (the circumflex and the horn) also use static hand pose, while the breve and 5 tonal symbols require hand motion to describe. In this study, we will consider only 23 static letters with 2 static additional accent marks, the other dynamic symbols will not be mentioned.

3.1.2. Data collecting

In this phase, we collected gesture images by photographing 35 participants, both male and female, aged 18 to 30. All participants were trained in the VSL standard and performed all 25 static symbols of the Vietnamese alphabet.

To capture their performances, we used various types of smartphone cameras to record videos of up to 20 minutes each. These videos were saved in MP4 and MOV formats with RGB encoding. The recordings employed different frame rates, including 24 FPS, 30 FPS, 60 FPS, and 120 FPS, to ensure diversity in the dataset.

To introduce variation in environmental conditions, we utilized multiple backgrounds with varying contrast and brightness levels. Additionally, participants positioned their hands at different distances from their bodies. Fig. 5 provides examples of these conditions. The dataset includes images from 30 individuals for training and 5 individuals for testing.

Fig. 5. Examples of hand images from our dataset

The data was captured under diverse conditions, including varying backgrounds, brightness levels, contrast settings, and distances between the hands and the body. This variability ensures that the dataset is robust and representative of real-world scenarios, helping the model generalize better during recognition tasks.

3.2. Data Pre-Processing

3.2.1. Video and image processing

In related works, techniques such as thresholding or binary mapping are often employed to segment the region of interest (ROI) or isolate hands in images [20]. Additionally, some methods simplify the process by converting images to grayscale [20]. In this paper, we introduce a novel preprocessing method that uses hand keypoints. Our approach detects 21 keypoints (hand landmarks) and reconstructs the hand shape by connecting these keypoints. This method leverages the OpenCV library [21] for image processing and the MediaPipe library [22] for hand detection and landmark extraction.

MediaPipe is highly effective for motion tracking and precise hand and finger recognition, thanks to its integration of machine learning (ML) models trained on extensive datasets by Google. By leveraging ML, the library can accurately estimate 21 hand landmarks with 3D coordinates from a single image. Furthermore, MediaPipe's efficient analytical framework enables faster and lighter processing, making it ideal for real-time applications on mobile devices.

The MediaPipe framework employs a palm detection model designed to enhance real-time usability on mobile platforms by utilizing a one-shot detector technique. For hand detection, MediaPipe uses two model variants: a reduced model and a full model, tailored to detect hands of varying sizes with magnifications of up to approximately 20x. Moreover, the model incorporates additional contextual information about the arm and the human body, enhancing contrast and improving hand recognition accuracy.

The detection process starts by identifying the palm region, which acts as the basis for detecting other parts of the hand. Each image is marked with 21 hand landmarks in 3D coordinates, later projected to 2D coordinates while omitting depth information. This training approach ensures high accuracy and robustness across diverse conditions, making MediaPipe well-suited for reliable hand recognition in real-time applications.

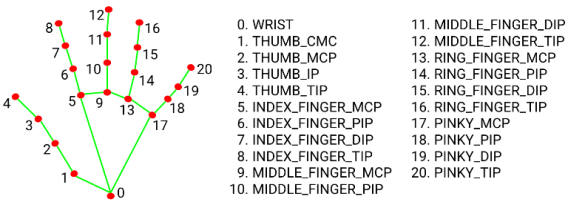

Fig. 6. 21 hand landmarks (keypoints)

Algorithm 1: Auto brightness and contrast balance method

1. input = image with size 512*512
2. input range = max(Input) - min(Input)
3. alpha = 255 / Input range
4. beta = -min(input) * alpha
5. output = alpha * input + beta

Before using the MediaPipe library to detect hands with 21 keypoints, we implemented a preprocessing algorithm to adjust brightness and contrast. This step improves the robustness of the MediaPipe library, ensuring it performs consistently and reliably under various background conditions.

Initially, the OpenCV library was utilized to process each frame of the video. A brightness and contrast balancing algorithm was applied to each frame to ensure consistent illumination. Following the preprocessing step, the MediaPipe library was employed to detect the hands within the image. The region of interest (ROI), which contained the hand, was then cropped into a square bounding box and resized to 512×512 pixels.

Next, the resized image was reprocessed using the MediaPipe library to extract the 21 hand keypoints. These keypoints were used to reconstruct an image consisting solely of the detected hand markers and the connections between them (representing joints and bones). The reconstructed image was placed on a smooth white background to minimize visual noise and emphasize the hand structure. This approach significantly improved the contrast of the hand against the background and reduced interference from image noise.

Finally, to optimize data size and prepare for further processing, the pixel intensity values of the reconstructed image were normalized from the range [0.255] to [0.1]

Algorithm 2: Pre-processing algorithm of proposed method (8 steps)

Input: Images of various sizes extracted from videos.

Step 1: Apply an automatic brightness and contrast balance method.

Step 2: Use MediaPipe to detect the hand.

Step 3: Crop the region of interest (ROI) into a square bounding box around the hand.

Step 4: Resize the cropped image to (512, 512).

Step 5: Normalize a copy of the resized image to the range (0, 1).

Step 6: Use MediaPipe to detect hand joints in the resized image.

Step 7: Draw the detected joints on a (512, 512) white background image using the coordinates from Step 6, and use this as the bone diagram image.

Step 8: Normalize the bone diagram image to the range (0, 1).

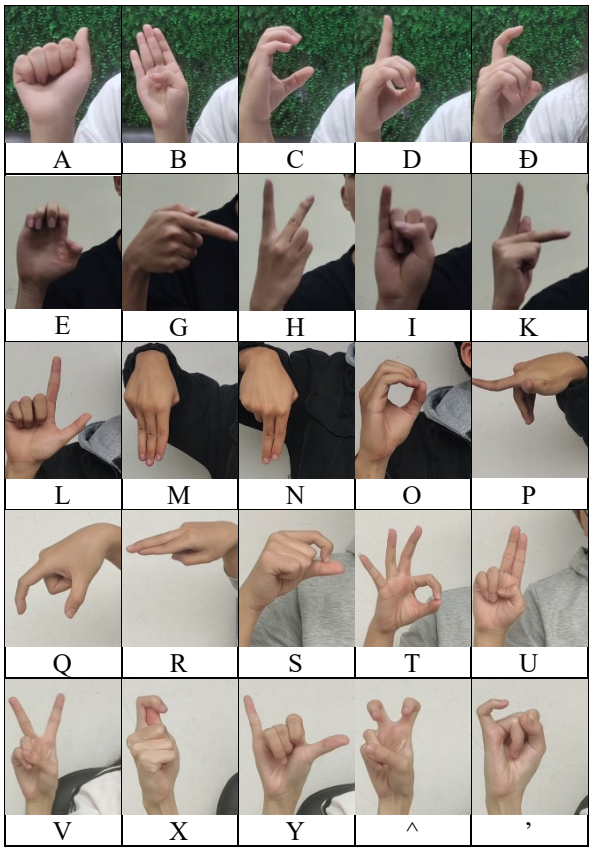

Fig. 7. A sample hand image alphabet of VSL after step 3

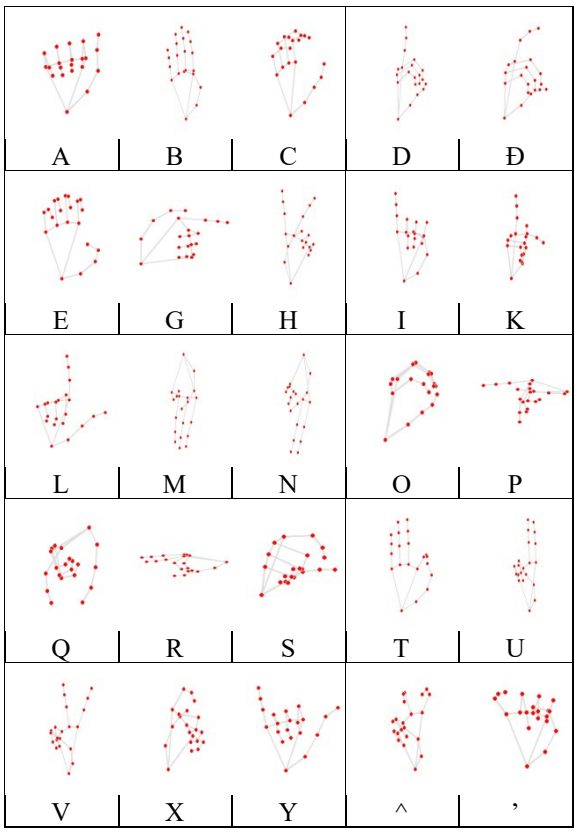

Fig. 8. A sample bone diagram alphabet of VSL after step 6

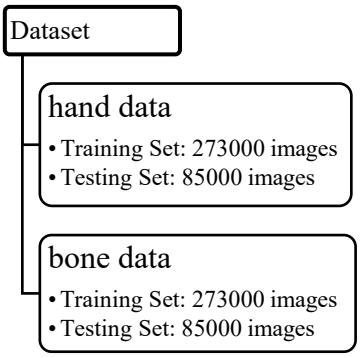

Fig. 9. Structure of the dataset, organized into two categories: "hand data" and "bone data"

After preprocessing the data, we obtained two related datasets, which we refer to as "hand data" and "bone data" (Fig. 9).

The "hand data" corresponds to the dataset 1 obtained after step 4, while the "bone data" refers to the dataset 2 generated after step 8. Each dataset includes a training set and a test set. The training set consists of approximately 270,000 images from 30 participants, while the test set contains nearly 85,000 images from 5 other participants. As a result, the "bone data" is significantly smaller in size compared to the "hand data.".

3.2.2. Data visualization

An additional step was taken to verify the suitability of our new data for use as a dataset in the machine learning process. By visualizing and evaluating the dataset, we assessed it based on several criteria: balanced class distribution, sufficient size, and an appropriate train-test ratio, as illustrated in Fig. 10, Fig. 11, Fig. 12.

The results indicate that our new dataset has balanced class sizes, with approximately 10,000 images in each training class folder and around 3,000 images in each test class folder. The train-test ratio of approximately 4:1 is considered typical for a machine learning dataset, ensuring a solid foundation for training and evaluation.

3.3. Deep Learning-Based Classification

3.3.1. Convolution neural network

CNN is one of the most used deep learning methods to analyse visual imagery. CNN involves less pre-processing compared to other image classification algorithms. The network learns the filters that are normally hand-engineered in other systems. The use of CNN reduces the images into a format that is easier to process while preserving features that are essential for making accurate predictions. There are four types of operations in a CNN: convolution, pooling, flattening, and fully connected layers [23] The convolution layer usually captures low-level features such as colour, edges, and gradient orientation. The pooling layer decreases the spatial dimension of the convolved feature. This operation reduces the required computational time for dealing with the data through dimensionality alleviation. Furthermore, it has the advantage of maintaining dominant features that are positionally and rotationally invariant during the model training process. After the input image has been processed the higher-level features may be used for classification. Therefore, the image is flattened into a 1-D vector. In CNN, the flattened output is supplied to a fully connected layer. After training, using SoftMax classification, the model can provide probabilities of prediction of objects in the image. Backpropagation is used to train the network. In this study, the system is implemented by using PyTorch library on GPU Zotac Gaming GeForce RTX 3090Ti.

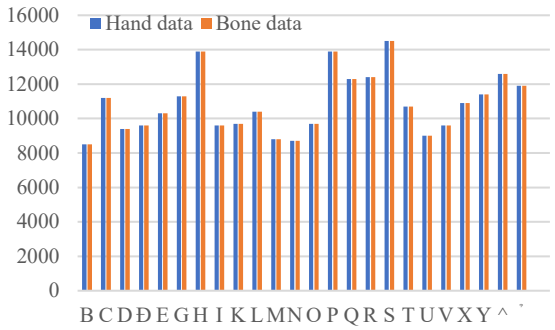

Fig. 10. Size of classes on training set of hand and bone data

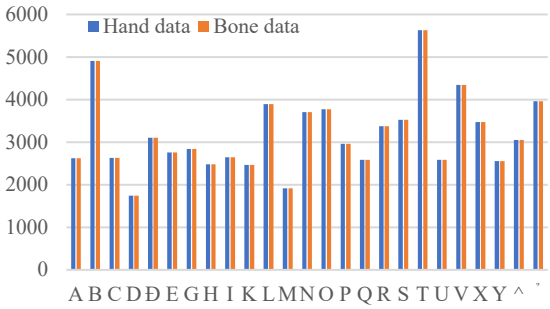

Fig. 11. Size of classes on testing set of hand and bone data

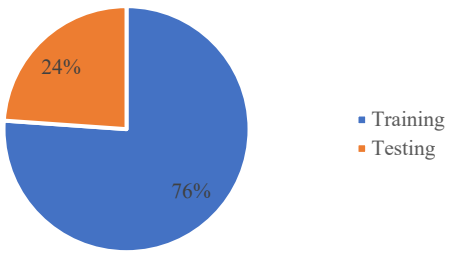

Fig. 12. Train-test ratio of hand data and bone data

3.3.2. The structure of the proposed convolutional neural network

The CNN model designed in our study consists of multiple layers. Fig. 13 illustrates the proposed structure of the CNN which consists of an input layer to input the images with 64 × 64 × 3 dimensions; this represents the size of the sign language frames that are taken as input into the system. The feature extraction part comprises three convolutional layers (Conv1, Conv2, Conv3). The convolution filter dimensions in each layer are 3 × 3. The batch normalization is 32 for ConvNet1, 64 for ConvNet2 and 128 for ConvNet3. Each convolution operation is followed by rectified linear units (ReLU). After ReLU, MaxPooling is applied. Pooling aims to prevent the loss of valuable information when the feature is represented. After the convolutional stage, flattening is applied for the classification stage. The classification stage is implemented with fully connected layers followed by a ReLU activation function and one SoftMax output layer.

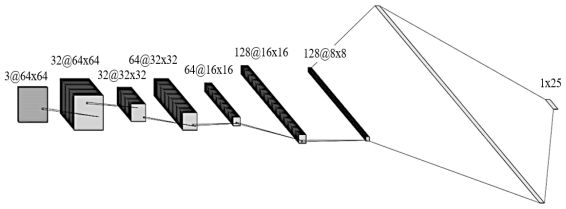

Fig. 13. Architecture of the proposed CNN model

3.4. Experimental Results

Since the proposed network is the simplified version of CNN, we calculate the loss of method by cross entropy loss function of PyTorch.

The training process was conducted using two distinct datasets: hand data and bone data (Table 3). Each dataset was used to train a separate model, denoted as Model I (Table 4) and Model II (Table 5), respectively. The models were evaluated using the cross-entropy loss function, a standard measure for classification tasks. Training was performed over 500 epochs, with an initial learning rate set to 0.001

Table 3. Result of testing model

Criteria \ Model	Model I	Model II
Sensitivity	0.9838	0.9634
Specificity	0.9970	0.9985
Precision	0.9244	0.9665
F1-score	0.9205	0.9637
Mean Accuracy	0.9208	0.9637

Table 4. Result of testing Model I on single label

	Pre	Rec	F1	Support
A	0.95	0.96	0.95	2620
B	0.96	0.98	0.97	4910
C	0.87	0.98	0.92	2632
D	0.87	0.87	0.87	1746
Đ	0.93	0.94	0.94	3102
E	0.94	0.93	0.94	2754
G	0.94	0.98	0.96	2842
H	0.65	0.97	0.78	2480
I	0.8	0.97	0.88	2647
K	1	0.85	0.92	2468
L	1	0.92	0.96	3892
N	0.97	0.93	0.95	1913
M	0.98	0.98	0.98	3706
O	0.92	0.86	0.89	3776
P	0.96	0.99	0.97	2958
Q	0.96	0.99	0.97	2584
R	0.98	0.98	0.98	3372
S	0.81	0.91	0.86	3527
T	0.96	0.83	0.89	5628
U	1	0.92	0.96	2586
V	0.88	0.98	0.93	4342
X	1	0.72	0.84	3474
Y	0.93	0.97	0.95	2557
^	0.89	0.82	0.85	3050
’	0.99	0.86	0.92	3959

Table 5. Result of testing Model II on single label

	Pre	Rec	F1	Support
A	0.98	0.95	0.97	2620
B	1	0.99	1	4910
C	0.98	0.99	0.99	2632
D	0.91	0.96	0.93	1746
Đ	0.99	0.91	0.95	3102
E	0.98	0.91	0.94	2754
G	0.99	1	1	2842
H	0.93	0.98	0.95	2480
I	0.97	0.97	0.97	2647
K	0.95	0.94	0.95	2468
L	1	0.95	0.97	3892
N	1	1	1	1913
M	1	0.99	0.99	3706
O	0.93	0.99	0.96	3776
P	0.99	0.96	0.97	2958
Q	1	1	1	2584
R	0.98	0.99	0.99	3372
S	0.98	1	0.99	3527
T	0.99	0.99	0.99	5628
U	0.98	1	0.99	2586
V	0.94	0.96	0.95	4342
X	0.99	0.77	0.86	3474
Y	0.97	0.98	0.98	2557
^	0.96	0.91	0.93	3050
’	0.79	0.98	0.88	3959

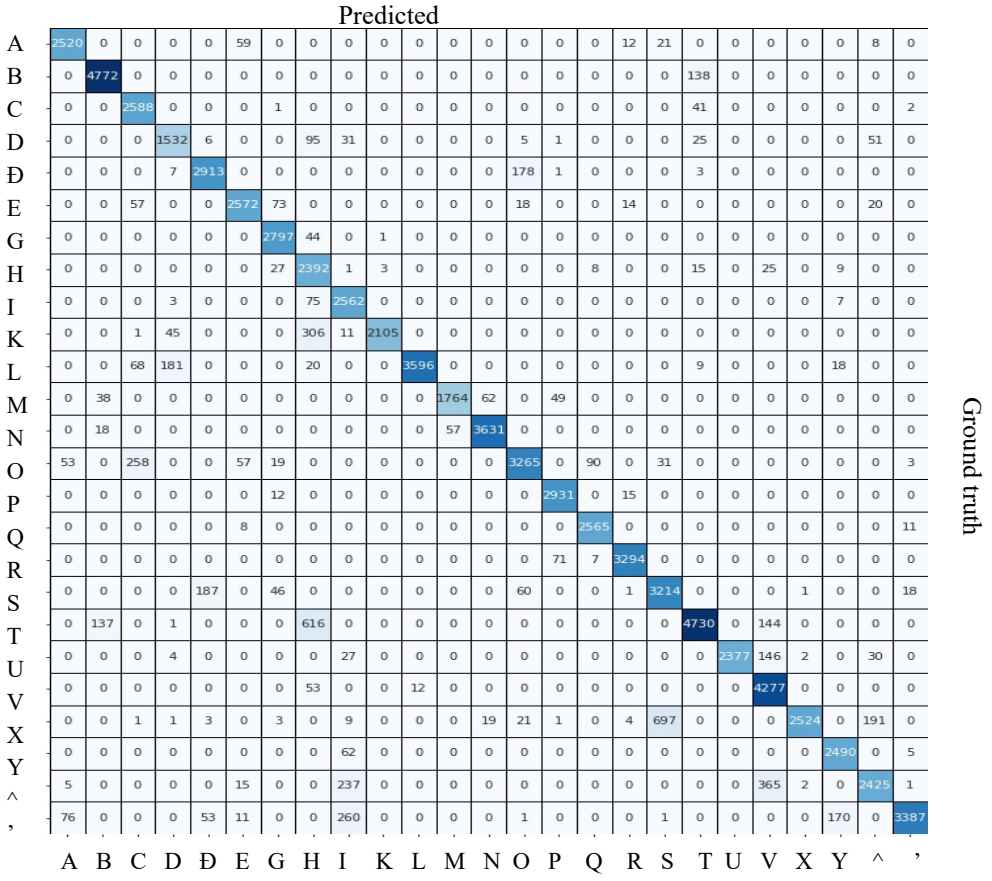

Fig. 14. Confusion matrix of trained model on test set of hand data

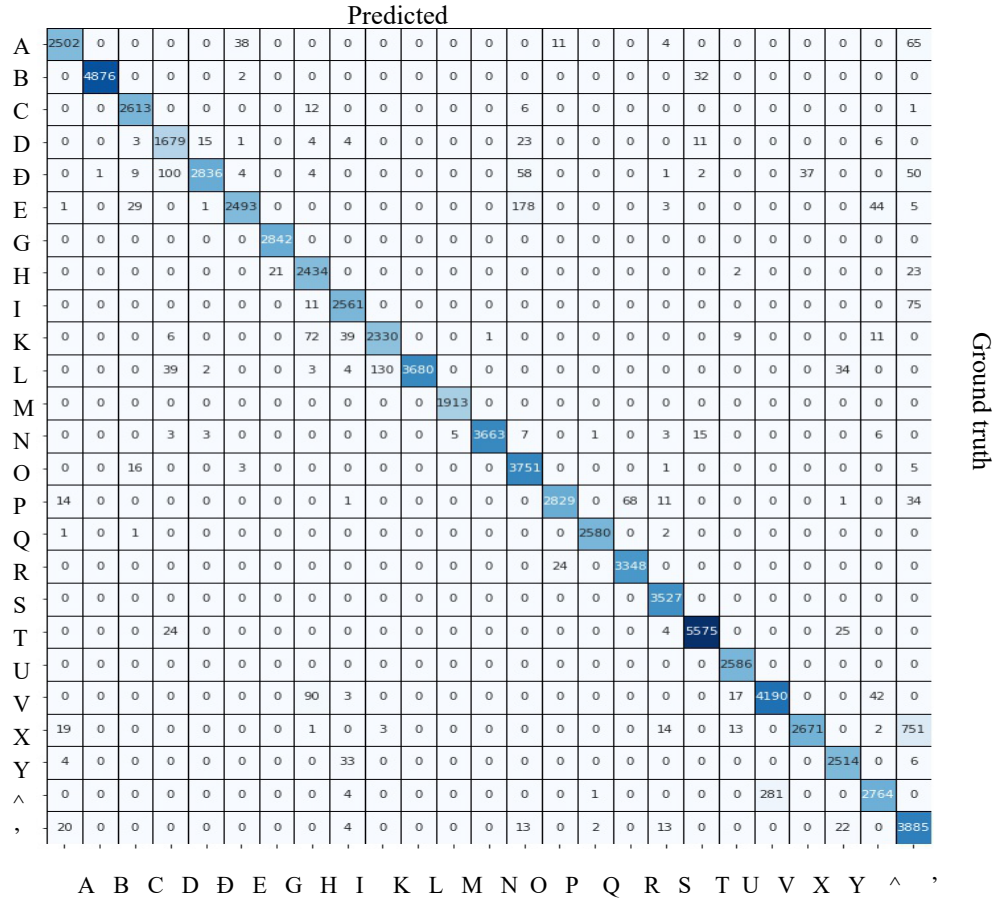

Fig. 15. Confusion matrix of trained model on test set of bone data

The learning rate decay was triggered when the training loss remained constant for two consecutive epochs, with the minimum learning rate capped at 0.0001. This adaptive reduction mechanism aimed to facilitate smoother convergence of the models.

Throughout the training process, the training loss of both Model I and Model II exhibited significant fluctuations in the initial epochs. However, the losses progressively decreased, reaching their minimum values around the 50th epoch. This reduction in training loss suggests that the models effectively learned the underlying features of the datasets. Notably, while the reduction in training loss indicated improved optimization, a slight decrease in training accuracy was observed, potentially signaling the onset of overfitting or challenges in further generalization.

Despite achieving accurate predictions in the majority of cases, the models demonstrated occasional failures. Upon detailed examination, these misclassifications were primarily attributed to alphabet similarities and the presence of difficult-to-recognize characters, which posed challenges for the model to distinguish effectively.

To evaluate the performance of the two trained models and validate our observations, we utilized the testing datasets of hand data and bone data to test Model I and Model II, respectively. The evaluation results are summarized in Tables 3, Table 4, and Table 5, and further illustrated by two confusion matrices in Fig. 14 and Fig. 15. The models were assessed using standard performance metrics, including sensitivity, specificity, precision, F1-score, and mean accuracy.

The evaluation results indicate that Model II outperformed Model I in terms of specificity, precision, F1-score, and mean accuracy. However, Model I achieved higher sensitivity compared to Model II. Additionally, the training time per epoch and response time per sample for Model II were observed to be lower than those of Model I, highlighting the computational efficiency of Model II.

4. Conclusion

Sign language serves as a vital communication tool for individuals with hearing and speech impairments. As such, sign language recognition plays a crucial role by capturing sign language videos and accurately interpreting the gestures. This paper focuses on VSL recognition, utilizing various local features and techniques for hand gesture identification.

The study centers on the development of a VSL recognition system capable of handling complex backgrounds, leveraging deep learning techniques. Specifically, the research utilizes a CNN model. The proposed methods were evaluated using a VSL dataset that includes the Vietnamese alphabet, incorporating unique characters such as “đ”, “”, and “^” with diverse samples collected from a large group of signers.

Experimental results demonstrate the effectiveness of the model, particularly when a novel pre-processing technique was applied prior to inputting data into the model. In particular, hand images were transformed into diagrams of hand shapes, marked with key points. Additionally, the pre-processing step proved useful in removing redundant background information, thereby improving the accuracy of the VSL recognition system. Future work will involve expanding the dataset and refining the pre-processing steps to enhance the recognition method, ultimately aiming to achieve higher accuracy.

Acknowledgments

This research is funded by Hanoi University of Science and Technology (HUST) under project number T2023-PC-028.

References

[1] R. J. Ruben, Sign language: Its history and contribution to the understanding of the biological nature of language, Acta Oto-Laryngologica, vol. 125, iss. 5, pp. 464-467, 2005. https://doi.org/10.1080/00016480510026287

[2] K. Emmorey, J. S. Reilly, and J. S. Reilly, Language, Gesture, and Space, New York, Psychology Press, 2013, 464 pp. https://doi.org/10.4324/9780203773413

[3] G. Plouffe and A.-M. Cretu, Static and dynamic hand gesture recognition in depth data using dynamic time warping, IEEE Transactions on Instrumentation and Measurement, vol. 65, iss. 2, pp. 305-316, Nov. 2015. https://doi.org/10.1109/TIM.2015.2498560

[4] S. Shah, A. Kotia, K. Nisar, A. Udeshi and P. P. M. Chawan, A vision based hand gesture recognition system using convolutional neural networks, International Research Journal of Engineering and dnhTechnology (IRJET), vol. 06, no. 04, Apr. 2019.

[5] A. Vo, B. N. Thiem and V. H. Pham, Deep Learning for Vietnamese sign language recognition in video sequence, International Journal of Machine Learning and Computing, vol. 9, no. 4, Aug. 2019. https://doi.org/10.18178/ijmlc.2019.9.4.823

[6] Z. Zhou, K. Chen, X. Li, S. Zhang, Y. Wu, Y. Zhou, K. Meng, C. Sun, Q. He, W. Fan, E. Fan, Z. Lin, X. Tan, W. Deng, J. Yang and J. Chen, Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays, Nature Electronics, pp. 571-578, Jun. 2020. https://doi.org/10.1038/s41928-020-0428-6

[7] Oyedotun, O.K., & Khashman, Deep learning in vision-based static hand gesture recognition, Neural Computing and Applications, vol. 28, iss. 12, pp. 3941-3951, Dec. 2017. https://doi.org/10.1007/s00521-016-2294-8

[8] D. Golekar, R. Bula, R. Hole, S. Katare and P. S. Parab, Sign language recognition using Python and

opencv, International Research Journal of Modernization in Engineering Technology and Science , vol. 04, iss. 02, pp. 1179-1183, Feb. 2022.

[9] Wang, H., Ru, B., Miao, X., Gao, Q., Habib, M., Liu, L., *et al*., MEMS devicesbased hand gesture recognition via wearable computing, Micromachines, vol. 14, iss. 5, Apr. 2023. https://doi.org/10.3390/mi14050947

[10] Wang, S., Wang, A., Ran, M., Liu, L., Peng, Y., Liu, M*., et al*, Hand gesture recognition framework using a lie group based spatio-temporal recurrent network with multiple hand-worn motion sensors, Information Sciences, vol. 606, pp. 722-74, Aug. 2022. https://doi.org/10.1016/j.ins.2022.05.085

[11] Al-Shamayleh, A. S., Ahmad, R., Abushariah, M. A. M., Alam, K. A., & Jomhari,A systematic literature review on vision based gesture recognition techniques, Multimedia Tools Applications, vol. 77, pp. 28121-28184, Apr. 2018. https://doi.org/10.1007/s11042-018-5971-z

[12] Rahman, M. M., Uzzaman, A., & Aktaruzzaman, M., Developing a real-time touchless human-computer interaction using hand gesture recognition, in IEEE CS BDC summer symposium. IEEE, Bangladesh, Jun. 2023.

[13] F. R. Cordeiro, S. Chevtchenko, R. F. Vale and V. Macario, A convolutional neural network with feature fusion for real-time hand posture recognition, Applied Soft Computing, vol. 73, pp. 748-766, Nov. 2018. https://doi.org/10.1016/j.asoc.2018.09.010

[14] P. Rathi, R. K. Gupta, S. Agarwal, A. Shukla and R. Tiwari, Sign language recognition using ResNet50 deep neural network architecture, in 5th International Conference on Next Generation Computing Technologies, Feb. 2020. https://doi.org/10.2139/ssrn.3545064

[15] P. Bhatia and A. Wadhawan, Deep learning-based sign language recognition system for static signs, Neural Computing and Applications, vol. 32, pp. 7957-7968, Jan. 2020. https://doi.org/10.1007/s00521-019-04691-y

[16] H. B. D. Nguyen and H. N. Do, Deep learning for american sign language fingerspelling recognition system in 2019 26th International Conference on Telecommunications (ICT), Hanoi, Vietnam, 2019, pp. 314-318. https://doi.org/10.1109/ICT.2019.8798856

[17] Bowen Shi, Diane Brentari, Greg Shakhnarovich, Karen Livescu; Fingerspelling detection in American Sign Language in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4166-4175, Jun. 2021 https://doi.org/10.1109/CVPR46437.2021.00415

[18] Costello, Elaine, American Sign Language Dictionary, Random House Reference, 2nd ed., 2008.

[19] Vietnamese Ministry of Education and Training, Promulgate regulations on national standards on sign language for people with disabilities, 2020.

[20] S. Chandran, Color image to grayscale image conversion, Conference on Computer Engineering and Applications (ICCEA), 2010 Second International Conference, vol. 2, Apr. 2010.

[21] N. Mahamkali and V. Ayyasamy, OpenCV for computer vision applications, Conference: Proceedings of National Conference on Big Data and Cloud Computing (NCBDC'15), March 20, 2015.

[22] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg and M. Grundmann, MediaPipe: A framework for building perception pipelines, arXiv preprint arXiv:1906.08172, Jun. 2019.

[23] Y. LeCun, Y. Bengio and G. Hinton, Deep learning, N U R S E, Nature, vol. 521, pp. 436-444, May. 2015. https://doi.org/10.1038/nature14539