Video-Based Sign Language Recognition via ResNet and LSTM Network

Jiayu Huang and Varin Chouvatut *

Posted Date: 29 May 2024

doi: 10.20944/preprints202405.1851.v1

Keywords: sign language recognition; deep learning; ResNet; LSTM

Article

Video-Based Sign Language Recognition via ResNet and LSTM Network

Jiayu Huang and Varin Chouvatut *

Department of Computer Science, Faculty of Scienc,Chiang Mai University,Chiang Mai 50200,Thailand
* Correspondence: varin.ch@cmu.ac.th

Abstract: Sign language recognition technology can help people with hearing impairments to communicate with those who are hearing impaired. At present, with the rapid development of society, deep learning technology also provided certain technical support for sign language recognition work. In sign language recognition tasks, the use of traditional convolutional neural networks to extract spatio-temporal features from sign language videos suffers from insufficient feature extraction, resulting in low recognition rates. Nevertheless, video-based sign language datasets are very large and require a lot of computational resources for training and generalisation must be ensured, which poses a challenge for recognition. In this paper, we have presented a video-based sign language recognition method based on resnet and lstm. As the number of network layers increases, the ResNet network can effectively solve the granularity explosion problem and obtain better time series features. We use the Resnet convolutional network as the backbone model. At the initialisation stage, we obtain sign language features using ResNet; then, the learned feature space is used as the input of LSTM network to obtain long sequence features. The experimental results show that the accuracy of the above model is better than the mainstream model, and it can effectively extract the spatio-temporal features in sign language videos and improve the recognition rate of sign language actions.

Keywords: sign language recognition; deep learning; ResNet; LSTM

1. Introduction

Sign language is a vital bridge for communication between deaf people and normal people. During communication, speakers will express their thoughts through various body movements and expressions, but few people in the sane population can understand sign language [1–3]; the goal of sign language recognition is to enhance the computer's understanding of human sign language, hence facilitating barrier-free communication between deaf and hearing people.

Sign language is a compound language that includes hand movements, body postures, and facial expressions to jointly display sign language vocabulary [4]. Most sign language meanings are explained by hand movements and body postures; Some emotional information can be expressed through facial expressions, such as using facial expressions to express joy, anger, sorrow, and happiness; Some can use facial expressions to distinguish the same gesture from expressing different meanings.

Video based sign language recognition can combine sign language as a continuous image frame with a temporal concept. The movement of gestures constitutes the basic component of sign language[5]. In the sign language library, it can be divided into static sign language and dynamic sign language based on its state [6]. Static sign language mainly includes basic letters and some basic traffic signs; It is represented by the shape and direction of hands and fingers, which constitute the basic actions of sign language words [7]. The dynamic sign language consists mainly of hand movements and facial expressions, and therefore requires video streaming to record these combinations of movements. All we do for sign language recognition is to recognise the sign language movements in the video stream.

There are different forms of sign language expression in different countries, and the correlation between sign language and spoken language is complex. The formation of sign language is to a large extent similar to the local living habits[8–10]. For the same language, different environments have

 doi:10.20944/preprints202405.1851.v1

different forms of sign language expression, such as American Sign Language (ASL) and British Sign Language (BSL), which are both in English but have different forms of sign language articulation. During sign language recognition tasks, the recognition methods are generic and require annotation of the dataset, followed by the use of algorithms to recognise and classify the data.

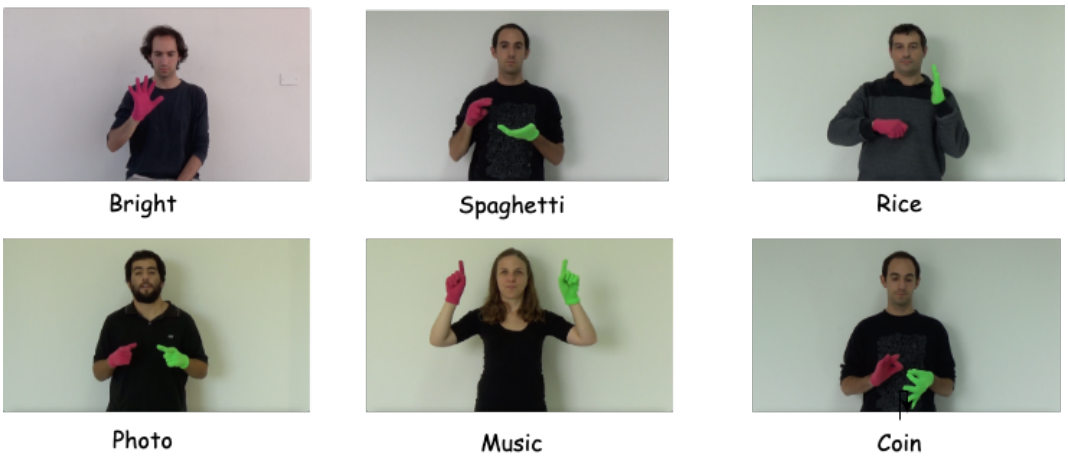

Figure 1. The image shows the keyframes extracted from Argentine sign language videos with different words, demonstrated with different signers to increase the richness and diversity of the data, making training more common. From the diagram, it can be seen that each performer is wearing fluorescent gloves, which better displays the movement trajectory of their hands during data collection, thereby increasing the recognizability of the data.

Traditional sign language recognition methods build temporal models by manually extracting features, and the temporal models used include Hidden Markov Models [11], Conditional Random Field [12], and Dynamic Time Warming [13] Manually feature extraction has been reliant on the experience of designers, and the process of temporal modelling is cumbersome, with no breakthroughs in many years. Researchers have used Convolutional Neural Networks (CNN) to exploit hand shape features with good results [14] Sign language words are made up of video sequences, and extracting features using 2D-CNN networks loses temporal information. Deep neural networks can extraction spatio-temporal features of videos, a significant breakthrough in behaviour recognition, which provides new insights for sign language recognition.

Although the methods mentioned above can perform sign language recognition, there are still some problems. With the increase of data volume and network layers, neural networks are prone to a series of factors that affect network stability, such as gradient explosion and gradient disappearance [15]; In time series, not only should the short-term feature space be considered, but the long-term feature space cannot be ignored. Long term time series can highlight global features and better capture the correlation between sequences in sign language recognition.

In this paper, we propose a framework for video-based sign language recognition via Resnet [16] and LSTM. The framework first uses the ResNet network to extract sign language features, obtains the sign language feature space, and then uses LSTM to obtain long-term sequence features. Finally, it uses a fully connected layer for classification, effectively completing sign language recognition. The notable contributions of this study can be summarized as follows:

- Word-level sign language recognition from video sequential files in Argentine sign language.
- A new end-to-end fusion Resnet and LSTM network is proposed for video based sign language sequence recognition. The network extracts key sign language features through residual networks, capture long-term relation of the video sequence, and achieves better classification performance.
- Pre-trained recognition method used in this research provides improvement in the recognition accuracy and training time in our model generation.

 doi:10.20944/preprints202405.1851.v1

- Motion recognition of the word-level sign language can be achieved with higher performance comparing with other methods as shown in comparison table in the results and discussion section.

The research on sign language recognition has been widely carried out both domestically and internationally, covering both theoretical and practical aspects. Many technical tasks. Sign language recognition based on different data processing methods technology can be divided into three research directions, summarized as follows.

2. Related Work

Using sensor devices to obtain gesture change signals and upper limb movement trajectories for modeling and achieving sign language translation. In 1983, Grimes et al. [17] were the first to use data gloves for sign language recognition research and achieved recognition of American sign language. Subsequently, more and more sign language recognition researchers are using data gloves for research on sign language recognition. Oliveira et al. [18] used two handed data gloves to capture sign language movements and fed them into neural networks for recognition, achieving recognition of English words. Lin et al. [19] used cameras to obtain data on people wearing colored gloves and performed data preprocessing such as color segmentation on these image data. Although sensor based sign language recognition has made significant progress, these devices require sign language performers to comply with specific wearing requirements, making the entire process cumbersome. Traditional methods such as image processing, sequence and classification algorithms are mainly used to achieve sign language recognition. Maharni et al. [20] proposed a gesture action classification system based on support vector machines. Liu et al. [21] used a K-value nearest neighbor method for gesture recognition, which measures the distance between different feature values for classification. Zhang et al. [22] proposed a model that combines DTW (dynamic time warping) and HMM (hidden Markov model) for recognizing continuous sign language videos. Experimental results show that this method can effectively reduce word error rates. Although traditional methods for sign language recognition have achieved certain results in accuracy, the limitations of manual computation and the complexity of gesture actions result in the use of manually set features greatly increasing the workload of sign language recognition. Therefore, more and more researchers are beginning to invest in sign language recognition based on deep learning.

Gesture change signals and upper limb movement trajectories are acquired using sensing devices to simulate and enable sign language interpretation. Koller et al. [23] achieved high recognition rates on the PHOENIX-2014 dataset by combining CNN (convolutional neural networks) with HMM for continuous sign language sentences. Considering the timing issue of sign language videos, Tran et al. [24] extended traditional 2D convolution to 3D convolution to obtain temporal features between video frames. Pigou et al. [25] captured hand features of the human body based on a CNN structure and constructed an Italian sign language recognition system, which achieved an accuracy of 91.7% for recognizing Italian sign language datasets. Cui et al. [26] used CTC to label time segments and combined CNN and RNN (current neural network) networks to improve the recognition rate of sign language videos, in order to extract advanced features from video time series information.

The Long Short-Term Memory (LSTM) network, as introduced by Hochreiter and Schmidhuber [27], represents a specialized Recurrent Neural Network (RNN) model distinguished by its unique structural design, adept at mitigating long-term dependency issues. Unlike conventional RNNs, LSTM inherently possesses the ability to retain early information, incurring no additional computational cost for this default behavior. LSTM achieves this by employing four distinct neural network layers that interact in a specialized manner. These layers are strategically designed to facilitate the learning of feature information in sequences. The incorporation of forgetting gates, memory gates, and output gates allows LSTM to selectively control the retention and transmission of sequence information. The cell state, serving as a repository of information, is utilized to store and transmit relevant data to subsequent LSTM units. This intricate process culminates in the reflection of the learned information into the cell state and output, enabling effective handling of sequential data with long-term

 doi:10.20944/preprints202405.1851.v1

dependencies. Mali et al. [28] used MediaPipe as a whole and LSTM modules to recognize sign language of people with disabilities. MediaPipe Holistic integrates pose, hand, and facial keypoints with precision levels, and is used due to its low latency and high tracking accuracy in real-world scenarios. It then uses LSTM modules for sign language classification and recognition. A dynamic sign language recognition method based on an improved LSTM model has been proposed by [29]. Using leapmotion to collect sign language, and then using LSTM network with attention mechanism for dynamic sign language recognition, achieved good results.

3. Methods

3.1. Notation

A sign language dataset with L labels training examples is denoted by $T_s = \{X_i, Y_i\}$,where $X_i \in \mathbb{R}^{C\times T\times H\times W}$, C is the channels of frame; T is the number of frames; H and W are the height and width of the frame respectively,and $Y_i \in \mathbb{R}^k$ is a label of K classes. We also consider a complementary source domain set of sign language data denoted by $D_s = \{D_s, M_i\}_i^N = 1$. Similarly, D_s is an RGB video. M_i represents the label sequence corresponding to D_i.

3.2. Overview

We have noticed that with the increase in the number of sign languages and the number of layers in deep learning networks, the model may suffer from gradient overfitting and exploding, which can lead to the robustness and recognition accuracy of the model; Time cost as a part of model training also needs to be considered; Long sequence features can better capture the correlation between time series. To address the aforementioned issues, we propose using a fusion network of ResNet and LSTM to train a sign language recognition model. As the network layers increase, the ResNet network can effectively solve the problem of gradient explosion and obtain better time series features. LSTM can obtain long sequence features. The specific process of this method is shown in Figure 2. We use Resnet convolutional network as the backbone model. In the initialization phase, we use ResNet to obtain sign language features; Then, the learned feature space was fed into the LSTM network as input to obtain long sequence features. Finally, classification output was performed and validated on the LSA64 dataset, achieving good recognition results.

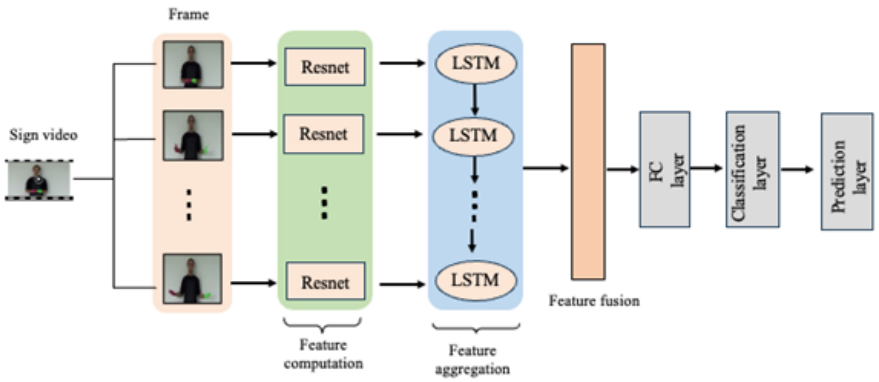

Figure 2. The flow is a process framework based on ResNet and LSTM. Firstly, the video data is preprocessed by dividing it into 16 frames of video segments. Then, each video segment is fed into the ResNet network for feature extraction to obtain a feature space. Secondly, in order to obtain long-term time series features, the learned feature space is fed into LSTM to obtain long-term features. Finally, the video is classified through a fully connected layer and the prediction results are obtained.

 doi:10.20944/preprints202405.1851.v1

3.3. Residual Convolution Network

The depth of the network is crucial to the performance of the model. When the number of network layers is increased, the network can extract more complex feature patterns. Therefore, theoretically, better results can be achieved when the model is deeper. However, is the performance of a deeper network necessarily better? The experiment found that deep networks have a degradation problem: as the depth of the network increases, the accuracy of the network saturates or even decreases. We know that deep networks have the problem of vanishing or exploding gradients, which makes it difficult to train deep learning models. But now there are some technological means such as BatchNorm to alleviate this problem. Therefore, the problem of deep network degradation is very surprising.

He et al. [16] proposed residual learning to solve the degradation problem. For a stacked layer structure (composed of several layers), when the input is, the learned features are denoted as. Now we hope that it can learn residuals, so that the original learned features are. The reason for this is that residual learning is easier than directly learning the original features. When the residual is 0, the stacked layer only performs identity mapping, at least the network performance will not decrease. In fact, the residual will not be 0, which will enable the stacked layer to learn new features based on the input features, thus having better performance. The structure of residual learning is shown in Figure 3. This is somewhat similar to a "short circuit" in a circuit, so it is a type of short connection. The residual unit can be represented as:

$$y_l = h(x_l) + F(x_l, Wl) \tag{1}$$

$$x_{l+1} = f(y_l) \tag{2}$$

Among them, and represent the input and output of the *th* residual unit, respectively. Note that each residual unit generally contains a multi-layer structure. It is a residual function that represents the learned residual, while it represents the identity mapping and is a ReLU activation function. Based on the above equation, we obtain the learning features from shallow to deep layers as follows:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i) \tag{3}$$

By using the chain rule, the gradient of the reverse process can be obtained.

$$\frac{\partial loss}{\partial x_l} = \frac{\partial loss}{\partial x_L} \cdot \frac{\partial x_L}{\partial x_l} = \frac{\partial loss}{\partial x_L} \cdot (1 + \frac{\partial}{\partial x_l} \sum_{i=l}^{L-1} F(x_i, w_i)) \tag{4}$$

The first factor $\frac{\partial loss}{\partial x_l}$ in the formula represents the gradient L reached by the loss function, while the 1 in parentheses indicates that the short-circuit mechanism can propagate the gradient without loss, while the other residual gradient needs to pass through a layer with weights, and the gradient is not directly transmitted. The residual gradient is not always -1, and even if it is relatively small, the presence of 1 will not cause the gradient to disappear. So residual learning will be easier.

 doi:10.20944/preprints202405.1851.v1

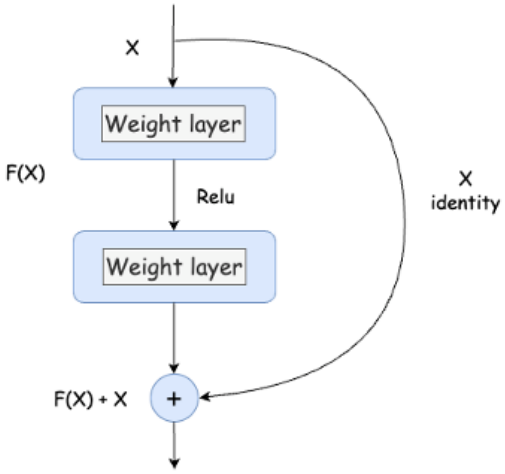

Figure 3. The diagram shows the resnet block, where the main branch of the residual structure consists of two layers of 3×3 convolutional layers, and the connecting line on the right side of the residual structure is the shortcut branch. The output matrix on the branch and the output matrix on the shortcut branch have the same shape

By using residual modules to increase network depth, the neural network in sign language video recognition tasks can retain both low-level and deep features without causing excessive repetitive learning. This can obtain the optimal feature representation to improve the accuracy of sign language recognition and the representation of semantic information. For sign language video recognition models, excessively deep network layers may focus too much on details and overlook the overall picture. Therefore, this article uses ResNet18 as the sign language feature extraction model.

3.4. LSTM Network

The sign language recognition task requires attention to the spatial and temporal features of the video. ResNet can extract a large amount of useful spatial feature information, but it has some shortcomings in extracting temporal feature information. The RNN model can effectively handle variable length data and model it, with a natural time depth for extracting temporal features. However, traditional RNN structures may cause network gradients to disappear or explode during model training due to long time spans, while sign language recognition tasks require long-term dependence on network timing. To address these issues, this article adopts LSTM networks for temporal modeling of sign language recognition. The LSTM network consists of three gates, an activation function, and a memory unit.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 May 2024 doi:10.20944/preprints202405.1851.v1

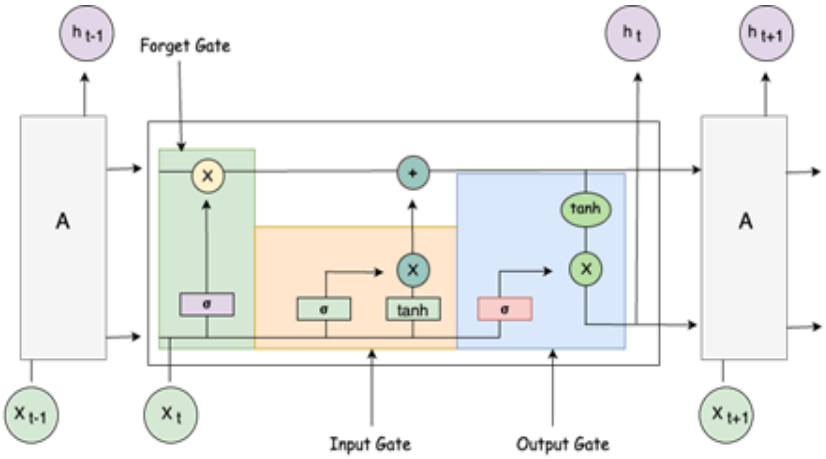

Figure 4. The diagram shows the structural information of LSTM, where C represents the cell state and is used to store the current state information and transmit it to the next LSTM. x represents the output information of the previous time, which is the output gate. f_t represents the forgetting gate, i_t represents the memory gate, and C_t represents the information that needs to be updated.

where σ is the sigmoid function, $tanh$ is a hyperbolic tangent function. Forgetting Gate f_t indicates how much information needs to be discarded and saved in the previous moment, while the remaining useful information is used at the current moment to handle the problem of gradient vanishing and exploding. Input gate i_t is used for filtering new memory expressions by discarding unnecessary information and retaining new useful information. Add the memory retained at the previous time in the network to the memory retained at the current time to obtain a new memory.The $tanh$ layer g_t generates a set of candidate values, which will be added to the storage unit if the input gate allows. According to (7), update the storage unit c_t based on the output of the forget gate f_t, input gate i_t, and new candidate value g_t. In formula (8), the output gate o_t controls the state and memory information of the hidden state. Finally, the hidden state is represented as the product of the storage unit state and the function of the output gate.

$$f_t = \sigma(W_{xr} * x_t + W_{hr} * H_{t-1}^k + b_r) \tag{5}$$

$$i_t = \sigma(W_{xi} * x_t + W_{hi} * H_{t-1}^k + b_i) \tag{6}$$

$$g_t = tanh(W_{xg} * x_t + W_{hg} * H_{t-1}^k + b_g) \tag{7}$$

$$o_t = \sigma(W_{xo} * x_t + W_{ho} * H_{t-1}^k + b_o) \tag{8}$$

$$c_t = f_t c_{t-1} + i_t g_t \tag{9}$$

$$h_t = o_t tanh(c_t) \tag{10}$$

3.5. Pre Training

In this section, we use pre trained models to improve training time, and train on the ImageNet dataset using Renest. we set the network to 18 layers and perform scratch training on the same input. The network parameters are shown in Table 1, which consists of 5 convolutional layers. The extracted video frames have a size of 128×171, and each frame is randomly cropped to 112×112. In this model training, the length of the video continuous sequence is set to $T = 16$, and each video clip contains 16 video frames. Use Batch normalization in each layer and set the batch size to 32. Random Gradient Descent (SGD) is set as the optimizer, the learning rate is set to 0.01, and every 10 epochs is divided by 10. Under the above settings, initialize and pre train the ResNet 18 layer convolutional network to obtain a pre trained model.

 doi:10.20944/preprints202405.1851.v1

4. Experimental Results

To verify the effectiveness of the proposed framework, we trained on the dataset and used different metrics as metrics to highlight performance. The experimental platform adopts the Ubuntu 18.04 system, and the algorithm model is built based on the open-source deep learning framework PyTorch. Hardware is Intel (R) Xeon (R) CPU E5-2620 v4@2.10 GHz CPU, 16 GB RAM, NVIDIA GeForce GTX 1080 Ti GPU 4 *times*4. Python 3 is a programming language used.

4.1. Video Pre-Processing

In the framework proposed in this article, video is a time series that is not used as raw input for direct input. In the early stage, video data needs to be processed. In the video sequence, we segment the video into segments, each containing 16 key image frames. In order to increase its richness, we randomly crop and batch normalize keyframes, which can eliminate other redundant information and better express the extracted features. RGB is the most intuitive mode, and we adjust the keyframe size to 128×128, with each video clip consisting of a 16 frame sequence. The size of the video clip is $16 \times 128 \times 128$.

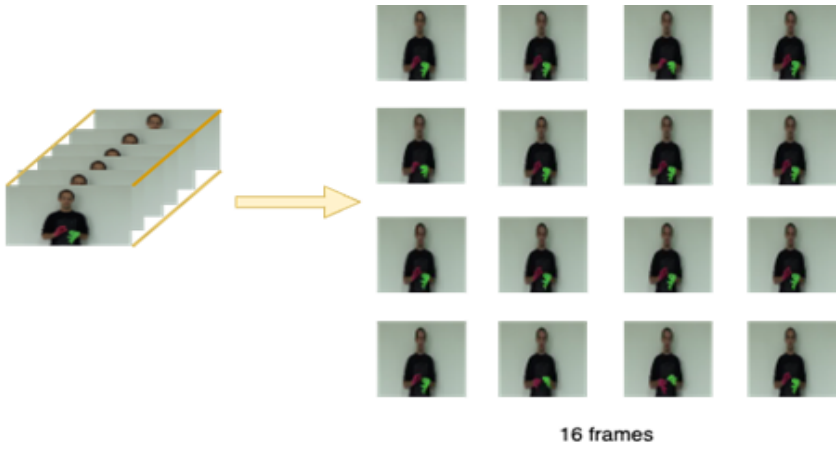

Figure 5. The diagram illustrates the process of data processing

4.2. Argentinie Sign Language Dataset

This experiment used the Argentine Sign Language dataset (LSA64) collected by the National University of La Plata [30]. LSA64 contains 64 categories of daily sign language words, each recorded by 10 participants, with each participant recording 5 times. The entire sign language dataset contains 3200 videos. Symbols are selected as the most commonly used symbols in LSA dictionaries, including verbs and nouns. In order to better demonstrate the trajectory characteristics of sign language, each performer wears pink gloves on their right hand and fluorescent gloves on their right hand. Some sign language is performed with the left hand, some with the right hand, and some with both hands simultaneously. In order to verify the generalization ability of the algorithm for non-specific individuals and compare it with other methods, this paper selected samples from 8 sign language learners as the training set, and the remaining 2 foreign language learners as the test set. Figure 1 shows the sample of LSA64 [31].

4.3. Metrics

In the experimental classification results, accuracy is one of the most common classification evaluation indicators to measure the accuracy of the classifier, which refers to the proportion of correctly classified samples to the total sample. In this adaptation, we used accuracy as the measurement indicator, but during the training process, we also obtained F1 score, precision, and Accuracy, which

 doi:10.20944/preprints202405.1851.v1

require four results to calculate, including true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The accuracy can be obtained by using the following formula:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{11}$$

4.4. Implementation Details and Parameters

In order to obtain better classification results and improve its training efficiency, the pre trained model mentioned above is used to freeze the weights of the pre trained model. After obtaining the feature space, the features are fed into the LSTM network to better train sign language videos. For specific setting parameters, in the selection of video clips, in order to better compare with other methods and consider the temporal continuity of sign language words, we selected a 16 frame sequence as the data input for each training session. In order to reduce training consumption and overfitting issues, each frame is randomly cropped to 128×128, and the training and testing sets have the same input size to better achieve test classification results. ReLU is used as a hidden activation function. The batch size of the network is set to 8, 16, and 32. Use Adam to optimize the model in the optimizer selection, with an initial learning rate of 0.0001 and weight decay of 0.0005. Using adaptive learning rate adjustment method [32] (Reduce LR On Plateau) to automatically change the learning rate; Set its optimal threshold to 0.0001, set the epoch with no improvement in tolerance indicators to 5, When designing the loss function, in order to prevent overfitting, we used label smoothing cross loss function to mitigate the impact of incorrect labels. The iteration period for each experiment is 50. The parameters are simplified as shown in Table 2:

Table 1. Training parameters

Parameters	**Setting**
Framework	Pytorch
Epochs	50
Batch-size	16
Frame size	128×128
Number of frames extracted per video	16
Number of feature extracted per frame	RGB
Optimizer	SGD

In this section, our proposed framework was validated on the LSA64 dataset, we divided the dataset into training and validation datasets. The training and validation datasets are independent sign language performers. We set different batch sizes of 8, 16, and 32, and used F1 score, precision, and accuracy measurement functions to obtain the accuracy of the validation set. Table 3 shows the results, and we can see from the table that the batch size setting has little effect on its training convergence effect. As the number of epochs increases, its training accuracy gradually stabilizes, reaching a stable value around 20 epochs. In addition, we observed that setting 30 epochs can also achieve good results.

 doi:10.20944/preprints202405.1851.v1

Table 2. The Validation Accuracy for different epoches and batch size on LSA64 Dataset.

Epochs	Batch Size	F1-Score	Precision	Accuracy
10	8	69.81%	76.30%	72.66%
10	16	75.66%	82.79%	77.50%
10	32	72.64%	77.13%	75.31%
20	8	81.28%	84.59%	82.97%
20	16	83.01%	86.61%	84.38%
20	32	78.11%	80.68%	80.47%
30	8	81.19%	83.88%	83.12%
30	16	84.98%	87.77%	**86.25%**
30	32	79.29%	82.07%	81.88%
40	8	81.98%	85.37%	84.38%
40	16	83.20%	96.10%	84.69%
40	32	81.13%	85.89%	82.81%
50	8	80.60%	83.94%	82.19%
50	16	83.89%	85.95%	85.94%
50	32	82.19%	83.22%	84.22%

Figure 6 shows the training process of the Argentine Sign Language dataset. The loss can indicate the convergence speed and stability of the model. We compared the training loss and validation loss of batches 8 and 16 at 30 epochs, respectively. From the figure, it can be seen that the fusion of pre train resnet and LSTM network framework has a good convergence efficiency in the recognition process. It gradually stabilizes at 23 epochs, with a training loss value of around 0.8 and a validation loss value of around 1.0. The reason for this situation is that the number of verified sign language datasets is too small. There was no vanishing gradient during the training process, and the use of the pre train model better demonstrated that our proposed method can reduce training time while ensuring recognition accuracy, thereby ensuring good robustness of the model.

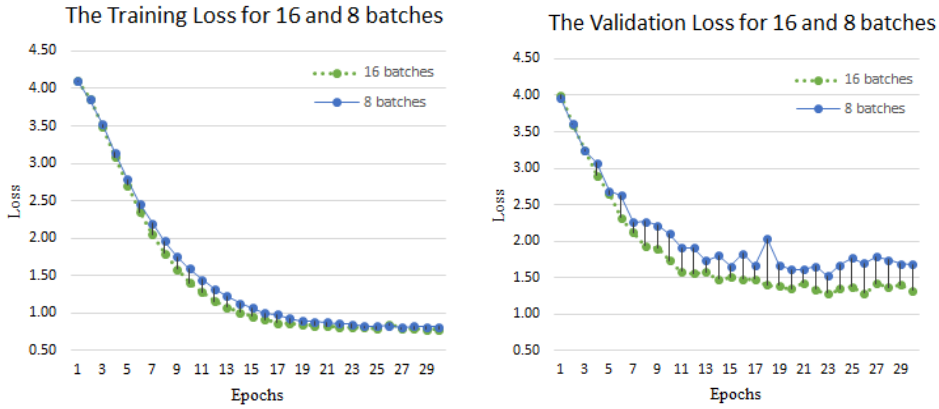

Figure 6. Training loss and validation loss on LSA64 Dataset. The left shows that the loss of training gradually decreases with the increase of iterations. The rihgt is loss value of validation gradually decreases with the increase of epochs

In Figure 7, we compared the training accuracy and validation accuracy, and the accuracy better reflects the model efficiency and usability. During the training process, the training accuracy gradually approached 100%, and the model stabilized at 19 epochs. In the subsequent training process, the training model showed fitting phenomenon, and the model achieved a recognition rate of 100%;

 doi:10.20944/preprints202405.1851.v1

During the validation training process, the validation accuracy of 16 batches was higher than that of 8 batches, reaching a maximum of 86.25%. The reason for this situation is largely due to the input data. We divided each video into 16 frames of video segments, and during the training process, 16 batches were also grouped into one epoch. This network design can better extract features, but from Figure 7, the amplitude of its variation is within a controllable range, which has little impact on recognition efficiency.

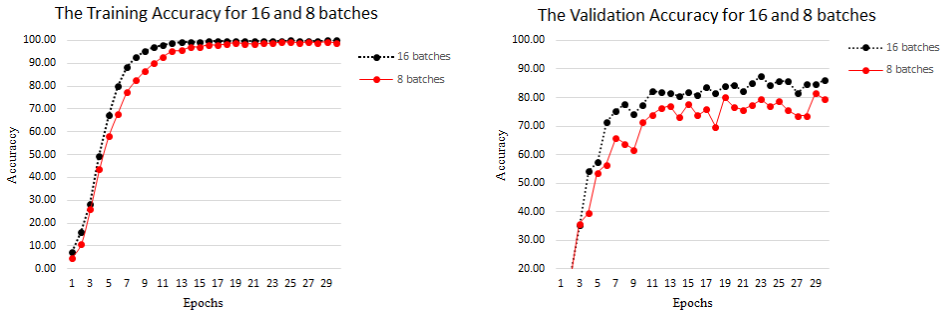

Figure 7. Training accuracy and loss on LSA64 Dataset. The left shows that the accuracy of training gradually increases with the increase of iterations. The rihgt is loss value of training gradually decreases with the increase of iterations

Table 3. Comparison of accuracy of different methods on LSA64 Dataset

Method	Modality	Classifier	Acc
3DCNN [33]	RGB	3DCNN	63.4%
Ensemble KNN [34]	RGB	KNN	64.0%
CNN-LSTM [35]	RGB	CNN and LSTM	72.4%
Logistic Regression [36]	RGB	Logistic Regression	73%
ResNet_LSTM(ours)	RGB	ResNet and LSTM	**86.25%**

Comparing the method used in this article with other methods, as shown in Table 4, it can be seen. For the 3DCNN model, although adding one-dimensional space for feature extraction in the time dimension, the effect is not very significant due to the fact that the 3DCNN model is based on continuous image convolution operation in processing time series. [34] proposes using the ensemble k-nearest neighbor algorithm for recognition, but its effectiveness is not very obvious. In the CNN-LSTM model, CNN is a simple neural network structure that cannot capture finer feature changes compared to residual network structures. [35] logistic regression algorithm was used for sign language recognition, which is a relatively easy classification prediction algorithm. However, in sign language recognition with time series, its performance only achieved a recognition rate of 73%.

After analysis, although the CRNN model composed of traditional CNN and LSTM can extract spatiotemporal features from sign language videos, the accuracy of sign language video recognition is affected by the vanishing or exploding gradients that traditional CNN may experience as the network hierarchy deepens. So in this article, using ResNet 18 instead of traditional CNN can not only solve the problem of vanishing and exploding network gradients, but also extract deeper feature information to improve the accuracy of network recognition. In addition, in order to better extract spatial feature information of sign language and improve model performance, this paper uses LSTM to learn long-term features of time series. The network can remove redundant information, thereby improving the accuracy and generalization ability of the model. In the end, we achieved an accuracy rate of 86.25% on the dataset, which better demonstrates the effectiveness of our proposed method.

5. Conclusions

On the basis of implementing video classification using two-dimensional networks, a series of factors that affect the stability of neural networks, such as gradient explosion and vanishing, are

 doi:10.20944/preprints202405.1851.v1

addressed; In time series, not only should the short-term feature space be considered, but the long-term feature space cannot be ignored. Long term time series can highlight global features and capture the correlation between sequences. This article proposes a dynamic video sign language recognition method based on the fusion network of Resnet and LSTM. The Resnet network is used as the skeleton model to learn the deep level feature information of dynamic sign language. LSTM can obtain feature information of long-term sequences and eliminate some redundant information. In the framework of this article, we also utilize the advantages of pre trained models to improve training efficiency and further enhance generalization in sign language recognition tasks. Finally, the performance of the model was validated on the LSA64 dataset, and the results showed that the accuracy and reliability of the model were high, verifying the usability and effectiveness of the modified model. Although the performance of the model in this article is good. Future work can focus on how to better learn spatial features, such as adding attention mechanisms and capturing the correlation of video sequences; In addition, multi pose fusion feature extraction is also worth studying.

Author Contributions: Conceptualization, H.J.; methodology, H.J.; Software, H.J.; Validation, H.J.; formal analysis, C.V.; Investigation, H.J. and C.V.; Resources, H.J.; Data curation, H.J. and C.V.; writing—original draft preparation, H.J. and C.V.; writing—review and editing, H.J. and C.V.; visualization, H.J. and C.V.; supervision, C.V.; project administration, H.J. and C.V. All authors have read and agreed to the published version of the manuscript.

Funding: Not applicable.

Acknowledgments: This research was supported by Fundamental Fund 2024, Chiang Mai University (FF030/2567).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Experimental data is available in this paper at https://facundoq.github.io/datasets/lsa64/.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Hu, H.; Zhou, W.; Pu, J.; Li, H. Global-Local Enhancement Network for NMF-Aware Sign Language Recognition. *ACM Trans. Multimedia Comput. Commun. Appl.* **2021**, *17*. doi:10.1145/3436754.
2. Huang, J.; Zhou, W.; Li, H.; Li, W. Attention-Based 3D-CNNs for Large-Vocabulary Sign Language Recognition. *IEEE Transactions on Circuits and Systems for Video Technology* **2019**, *29*, 2822–2832. doi:10.1109/TCSVT.2018.2870740.
3. Huang, J.; Zhou, W.; Zhang, Q.; Li, H.; Li, W. Video-Based Sign Language Recognition without Temporal Segmentation. Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence. AAAI Press, 2018, AAAI'18/IAAI'18/EAAI'18.
4. Das, S.; Biswas, S.K.; Chakraborty, M.; Purkayastha, B. Intelligent Indian Sign Language Recognition Systems: A Critical Review. ICT Systems and Sustainability; Tuba, M.; Akashe, S.; Joshi, A., Eds.; Springer Nature Singapore: Singapore, 2022; pp. 703–713.
5. Cheok, M.J.; Omar, Z.; Jaward, M.H. A review of hand gesture and sign language recognition techniques. *International Journal of Machine Learning and Cybernetics* **2019**, *10*, 1–23.
6. Yang, S.; Zhu, Q. Video-based Chinese sign language recognition using convolutional neural network. 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), 2017, pp. 929–934. doi:10.1109/ICCSN.2017.8230247.
7. Chouvatut, V.; Panyangam, B.; Huang, J. Chinese Finger Sign Language Recognition Method with ResNet Transfer Learning. 2023 15th International Conference on Knowledge and Smart Technology (KST), 2023, pp. 1–6. doi:10.1109/KST57286.2023.10086825.
8. Makhashen, G.M.B.; Luqman, H.A.; El-Alfy, E.S.M. Using Gabor filter bank with downsampling and SVM for visual sign language alphabet recognition. 2nd Smart Cities Symposium (SCS 2019), 2019, pp. 1–6. doi:10.1049/cp.2019.0188.

9. Madhiarasan, M.; Roy, P.P. A Comprehensive Review of Sign Language Recognition: Different Types, Modalities, and Datasets. *ArXiv* **2022**, *abs/2204.03328*.
10. Pu, J.; Zhou, W.; Li, H. Dilated Convolutional Network with Iterative Optimization for Continuous Sign Language Recognition. Proceedings of the 27th International Joint Conference on Artificial Intelligence. AAAI Press, 2018, IJCAI'18, p. 885–891.
11. Starner, T.; Weaver, J.; Pentland, A. Real-time American sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1998**, *20*, 1371–1375. doi:10.1109/34.735811.
12. Yang, H.D.; Sclaroff, S.; Lee, S.W. Sign Language Spotting with a Threshold Model Based on Conditional Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2009**, *31*, 1264–1277. doi:10.1109/TPAMI.2008.172.
13. Jangyodsuk, P.; Conly, C.; Athitsos, V. Sign language recognition using dynamic time warping and hand shape distance based on histogram of oriented gradient features. Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments; Association for Computing Machinery: New York, NY, USA, 2014; PETRA '14. doi:10.1145/2674396.2674421.
14. Köpüklü, O.; Gunduz, A.; Kose, N.; Rigoll, G. Real-time Hand Gesture Detection and Classification Using Convolutional Neural Networks. 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), 2019, pp. 1–8. doi:10.1109/FG.2019.8756576.
15. Chung, W.Y.; Xu, H.; Lee, B.G. Chinese Sign Language Recognition with Batch Sampling ResNet-Bi-LSTM. *SN Comput. Sci.* **2022**, *3*. doi:10.1007/s42979-022-01341-4.
16. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.
17. Grimes, G.J. Digital data entry glove interface device.
18. Oliveira, T.; Escudeiro, N.; Escudeiro, P.; Rocha, E.; Barbosa, F.M. The VirtualSign Channel for the Communication Between Deaf and Hearing Users. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje* **2019**, *14*, 188–195. doi:10.1109/RITA.2019.2952270.
19. Lin, Y.; Chai, X.; Zhou, Y.; Chen, X. Curve Matching from the View of Manifold for Sign Language Recognition. Computer Vision - ACCV 2014 Workshops; Jawahar, C.V.; Shan, S., Eds.; Springer International Publishing: Cham, 2015; pp. 233–246.
20. Maharani, D.A.; Fakhrurroja, H.; Riyanto.; Machbub, C. Hand gesture recognition using K-means clustering and Support Vector Machine. 2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2018, pp. 1–6. doi:10.1109/ISCAIE.2018.8405435.
21. Liu, Y.; Wang, X.; Yan, K. Hand gesture recognition based on concentric circular scan lines and weighted K-nearest neighbor algorithm. *Multimedia tools and applications* **2018**.
22. Zhang, J.; Zhou, W.; Xie, C.; Pu, J.; Li, H. Chinese sign language recognition with adaptive HMM. 2016 IEEE International Conference on Multimedia and Expo (ICME), 2016, pp. 1–6. doi:10.1109/ICME.2016.7552950.
23. Koller, O.; Zargaran, S.; Ney, H. Re-Sign: Re-Aligned End-to-End Sequence Modelling with Deep Recurrent CNN-HMMs. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3416–3424. doi:10.1109/CVPR.2017.364.
24. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning Spatiotemporal Features with 3D Convolutional Networks. 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4489–4497. doi:10.1109/ICCV.2015.510.
25. Pigou, L.; Dieleman, S.; Kindermans, P.J.; Schrauwen, B. Sign Language Recognition Using Convolutional Neural Networks. Computer Vision - ECCV 2014 Workshops; Agapito, L.; Bronstein, M.M.; Rother, C., Eds.; Springer International Publishing: Cham, 2015; pp. 572–578.
26. Cui, R.; Liu, H.; Zhang, C. Recurrent Convolutional Neural Networks for Continuous Sign Language Recognition by Staged Optimization. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1610–1618. doi:10.1109/CVPR.2017.175.
27. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Computation* **1997**, *9*, 1735–1780. doi:10.1162/neco.1997.9.8.1735.
28. Mali, P.; Shakya, A.; Panday, S.P. Sign Language Recognition Using Long Short-Term Memory Deep Learning Model. Fourth International Conference on Image Processing and Capsule Networks; Shakya, S.; Tavares,

J.M.R.S.; Fernández-Caballero, A.; Papakostas, G., Eds.; Springer Nature Singapore: Singapore, 2023; pp. 697–709.

29. Wu, B.; Lu, Z.; Yang, C. A Modified LSTM Model for Chinese Sign Language Recognition Using Leap Motion. 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2022, pp. 1612–1617. doi:10.1109/SMC53654.2022.9945287.
30. Ronchetti, F.; Quiroga, F.; Estrebou, C.; Lanzarini, L.; Rosete, A. LSA64: A Dataset of Argentinian Sign Language. In Proceedings of the Congreso Argentino de Ciencias de la Computación (CACIC), 2016.
31. Argentinian Sign Language Dataset. https://facundoq.github.io/datasets/lsa64/.
32. Zhang, Y.; Shen, L. Automatic Learning Rate Adaption for Memristive Deep Learning Systems. *IEEE Transactions on Neural Networks and Learning Systems* **2023**, pp. 1–12. doi:10.1109/TNNLS.2023.3244006.
33. Huang, J.; Zhou, W.; Li, H.; Li, W. Sign Language Recognition using 3D convolutional neural networks. 2015 IEEE International Conference on Multimedia and Expo (ICME), 2015, pp. 1–6. doi:10.1109/ICME.2015.7177428.
34. addin I. Sidig, A.; Mahmoud, S.A. Trajectory based Arabic Sign Language Recognition. *International Journal of Advanced Computer Science and Applications* **2018**, *9*.
35. Luqman, H.; El-Alfy, E.S.M. Towards Hybrid Multimodal Manual and Non-Manual Arabic Sign Language Recognition: mArSL Database and Pilot Study. *Electronics* **2021**, *10*. doi:10.3390/electronics10141739.
36. Sabyrov, A.; Mukushev, M.; Kimmelman, V. Towards Real-time Sign Language Interpreting Robot: Evaluation of Non-manual Components on Recognition Accuracy. CVPR Workshops, 2019.