Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN

Sarfaraz Masood, Adhyan Srivastava, Harish Chandra Thuwal and Musheer Ahmad

Abstract There is a need of a method or an application that can recognize sign language gestures so that the communication is possible even if someone does not understand sign language. With this work, we intend to take a basic step in bridging this communication gap using Sign Language Recognition. Video sequences contain both the temporal and the spatial features. To train the model on spatial features, we have used inception model which is a deep convolutional neural network (CNN) and we have used recurrent neural network (RNN) to train the model on temporal features. Our dataset consists of Argentinean Sign Language (LSA) gestures, belonging to 46 gesture categories. The proposed model was able to achieve a high accuracy of 95.2% over a large set of images.

1 Introduction

Sign language is a vision-based language which uses an amalgamation of variety of visuals like hand shapes and gestures, orientation, locality and movement of hand and body, lip movement and facial expressions. Like the spoken language, regional variants of sign language also exist, e.g., Indian Sign language (ISL), American Sign Language (ASL), and Portuguese Sign Language. There are three types of sign languages: spelling each alphabet using fingers, sign vocabulary for words, using hands and body movement, facial expressions, and lip movement. Sign language can also be isolated as well as continuous. In isolated sign language, people communicate using gestures of single word, while continuous sign language is a sequence of gestures that generate a meaningful sentence.

All the methods for recognizing hand gestures can be broadly classified as vision-based and based on measurements made by sensors in gloves. The vision-based method involves human and computer interaction for gesture recognition, while

S. Masood (✉) · A. Srivastava · H. C. Thuwal · M. Ahmad
Department of Computer Engineering, Jamia Millia Islamia, New Delhi 110025, India
e-mail: smasood@jmi.ac.in

V. Bhateja et al. (eds.), *Intelligent Engineering Informatics*, Advances in Intelligent Systems and Computing 695, https://doi.org/10.1007/978-981-10-7566-7_63

glove-based method depends on external hardware for gesture recognition. Significant works [1–6] in this field have been noted recently.

Ronchetti et al. [1] discussed an image processing based method for extraction of descriptor followed by a hand shape classification using ProbSom which is a supervised adaptation of self-organizing maps. Using this technique, they were able to achieve an accuracy of above 90% on Argentinean Sign Language.

Joyeeta and Karen [2] gave a method based on eigenvector. Skin filtering and histogram matching were performed in the preprocessing stage. The classification technique they used was based on eigenvalue-weighted Euclidean distance. They identified 24 different alphabets of Indian Sign Language with an accuracy of 96%.

Kumud and Neha [3] proposed a method for recognizing gestures from a video containing multiple gestures of Indian Sign Language. They extracted the key frame, based on gradient, to split the video to independent isolated gesture. The features were extracted from gestures by applying Orientation Histogram and Principal Component Analysis. Correlation, Manhattan, and Euclidean distance were used for classification and found that correlation and Euclidean distances gave better accuracies.

Anup et al. [4] demonstrated a statistical technique for recognizing the gestures of Indian Sign Language in real time. The authors created a video database for various signs. They used the direction histogram, which is invariant to illumination and orientation changes, as the feature for classification. Two approaches, Euclidean distance and K-nearest neighbor metrics, were used for gesture recognition.

Lionel et al. [5] proposed a system to recognize Italian sign language gestures. They used Microsoft Kinect and convolutional neural network (CNN) accelerated via graphic processing unit (GPU). They achieved a cross-validation accuracy of around 92% on a dataset consisting of 20 Italian gestures.

Rajat et al. [6] proposed a finely tuned portable device as a solution to alleviate this problem of minimizing the communication gap between normal and differently abled people. The architecture of the device, and its operations were discussed using three embedded algorithms which aimed for fast, easy, and efficient communication.

Another work by S. Masood et al. [7] have shown the use of Convolutional Neural networks for the purpose of character recognition for American Sign Language. In this work, the CNN based model was able achieve an overall accuracy of 96% on an image dataset of 2524 ASL gestures.

Also the creative works of the like of W. Vicars [8] have helped in general understanding in the field of American Sign Language recognition (Fig. 1).

In this work, we attempt to perform recognition on isolated sign language with the vision-based method. Unlike other works, we chose a dataset with larger gesture variants and significant video samples so that the resultant model having better generalization capabilities. In this work, we also attempt to explore the possibilities of exploiting the benefits of RNN in performing gesture recognition.

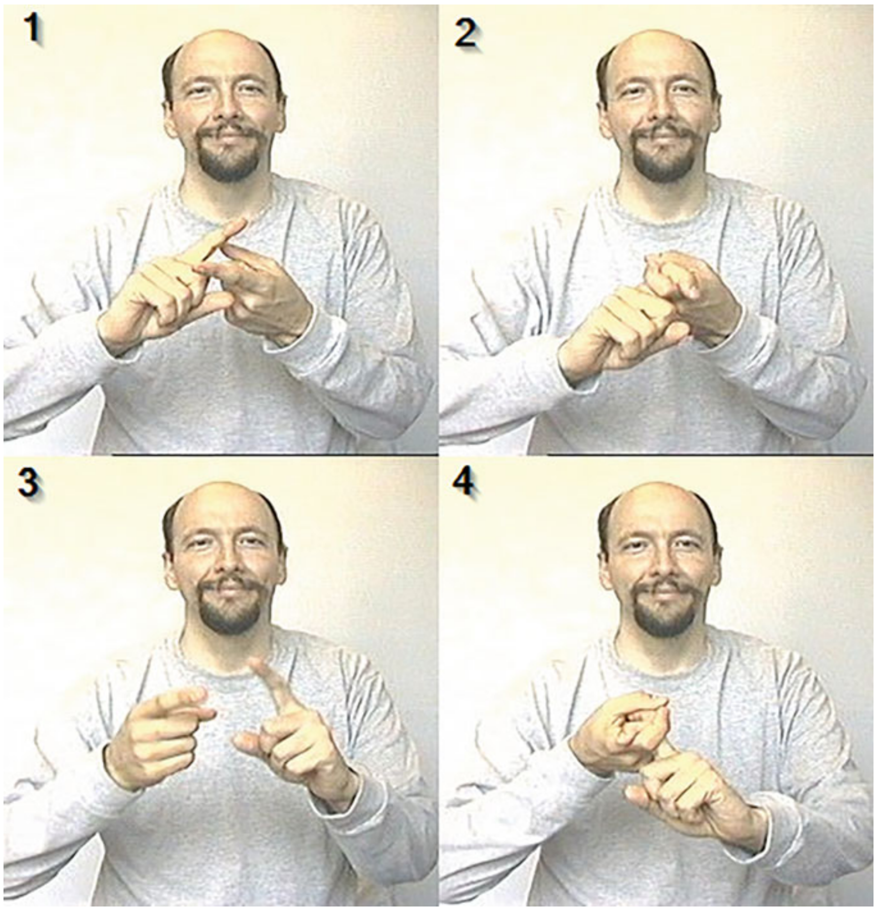

Fig. 1 ASL gesture for word “Friend” [8]

2 Algorithms Used

Video classification is a challenging problem as a video sequence contains both the temporal and the spatial features. Spatial features are extracted from the frames of the video, whereas the temporal features are extracted by relating the frames of video in a course of time. We have used two types of learning networks to train our model on each type of features. To train the model on spatial features, we have used CNN, and for the temporal features we have used recurrent neural network.

2.1 Convolutional Neural Network

Convolutional neural network or ConvNets are great at capturing local spatial patterns in the data. They are great at finding patterns and then use those to classify images. ConvNets explicitly assume that input to the network will be an image. CNNs, due to the presence of pooling layers, are insensitive to rotation or translation of two similar images; i.e., an image and its rotated image will be classified as the same image.

Due to the vast advantages of CNN in extracting the spatial features of an image, we have used Inception-v3 [9] model of the TensorFlow [10] library which is a deep ConvNet to extract spatial features from the frames of video sequences. Inception is a huge image classification model with millions of parameters for images to classify.

2.2 Recurrent Neural Network

There is information in the sequence itself, and recurrent neural networks (RNNs) use this for the recognition tasks. The output from an RNN depends on the combination of current input and previous output as they have loops. One drawback of RNN is that, in practice, RNNs are not able to learn long-term dependencies [11]. Hence, our model used Long Short-Term Memory (LSTM) [12], which is a variation of RNN with LSTM units. LSTMs can learn to bridge time intervals in excess of 1000 steps even in case of noisy, incompressible input sequences (Fig. 2).

The first layer is to feed input to the upcoming layers whose size is determined by the size of the input. Our model is a wide network consisting of single layer of 256 LSTM units. This layer is followed by a fully connected layer with softmax activation. Finally, a regression layer is applied to perform a regression to the provided input. We used Adaptive Moment Estimation (ADAM) [13] which is a

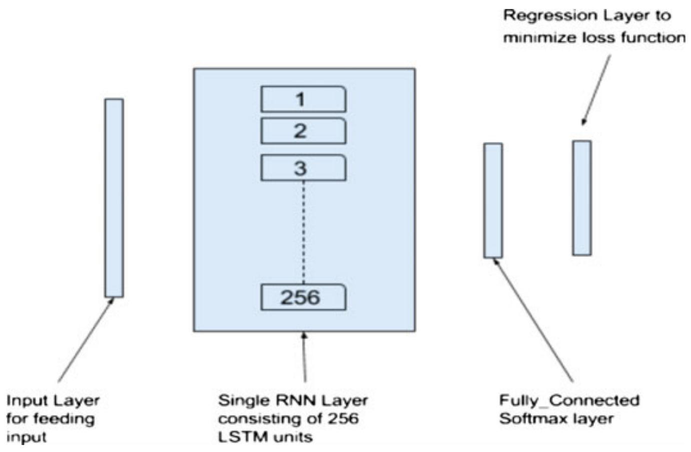
Regression Layer to minimize loss function
1
2
3
256
Input Layer for feeding input
Single RNN Layer consisting of 256 LSTM units
Fully_Connected Softmax layer

Fig. 2 Architecture of the proposed RNN model

stochastic optimizer, as a gradient descent optimizer to minimize the loss_function. Also, a wider RNN network was tried with 512 LSTM units and another deep RNN network with three layers of 64 LSTM units each. We tested these on a sample of the dataset and found that wide model with 256 LSTM units performed the best.

3 Methodology

Two approaches were used to train the model on the temporal and the spatial features, and both differ by the way inputs given to RNN to train it on the temporal features.

3.1 Prediction Approach

In this approach, spatial features for individual frames were extracted using inception model (CNN) and temporal features using RNN. Each video was then represented by a sequence of predictions made by CNN for each of their individual frames. This was given as input to the RNN. For every video corresponding to each gesture, frames were extracted and the background body parts other than hand were removed to get a grayscale image of hands which avoided color-specific learning of the model (Fig. 3).

Frames of the training set were given to the CNN model for training on the spatial features. The obtained model was then used to make and store predictions for the frames of the training and test data. The predictions corresponding to the frames of the training data were then given to the LSTM RNN model for training on the temporal features. Once the RNN model was trained, the predictions corresponding to the frames of the test data were fed to it for testing.

3.1.1 Train CNN (Spatial Features) and Prediction

Figure 4 depicts the role of CNN which is the inception model. From the training dataset, for each gesture “X”, all frames from each video corresponding to that gesture were labelled “X” and were given to the inception model for training.

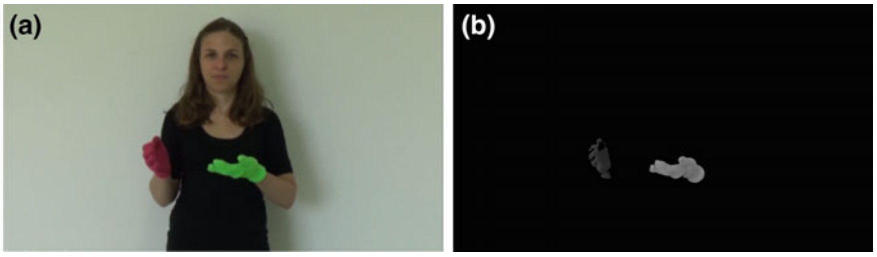

Fig. 3 **a** Sample frame from the dataset [14]. **b** Frame after background removal

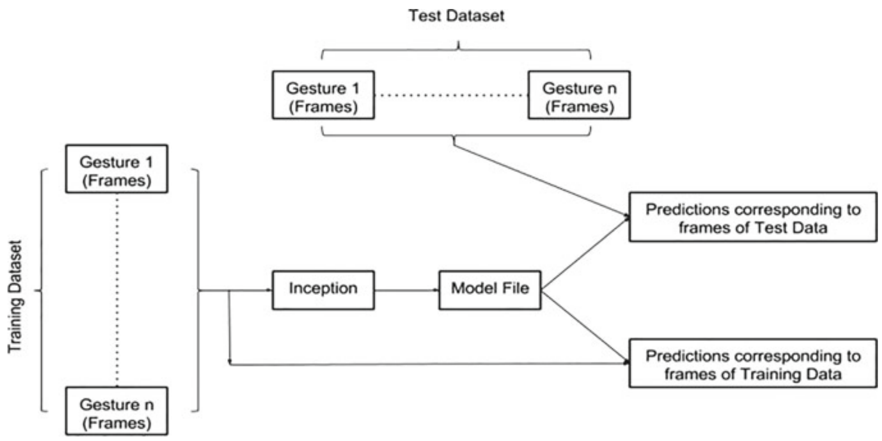

Fig. 4 Training CNN and saving prediction

The trained model was used to make predictions corresponding to frames of videos belonging to both the train and test set. This created two sets of predictions:

1. One corresponding to frames of training dataset—for training the RNN.
2. Another corresponding to frames of test dataset—for testing RNN.

Each gesture video was broken to a sequence of frames. Then after training CNN and making predictions, the video is represented as a sequence of predictions.

3.1.2 Training RNN (Temporal Features) and Testing

The videos for each gesture are fed to RNN as sequence of predictions of its constituent frames. The RNN learns to recognize each gesture as a sequence of predictions. After the Training of RNN completes a model file is created (Fig. 5).

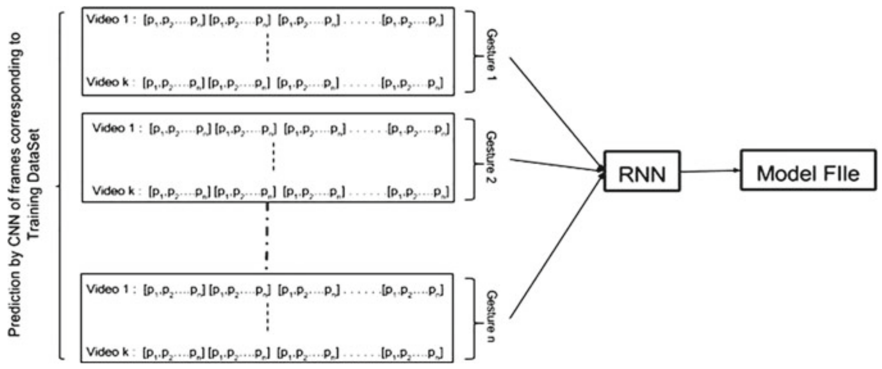
Prediction by CNN of frames corresponding to Training DataSet
Video 1 : [p1,p2....pn] [p1,p2....pn] [p1,p2....pn] ...... [p1,p2....pn]
Video k : [p1,p2....pn] [p1,p2....pn] [p1,p2....pn] ...... [p1,p2....pn]
Gesture 1
Gesture 2
Gesture n
RNN
Model File

Fig. 5 Training RNN after getting the results from CNN

The predictions of CNN for frames of the test set were fed to the trained model for testing. The model was used to recognize the sequence of predictions in the test set.

3.2 Pool Layer Approach

In this approach, CNN was used to train the model on the spatial features and passed the pool layer output to the RNN before it is made into a prediction. The pool layer gives a 2048-dimensional vector that represents the convoluted features of the image, but not a class prediction. Rest of the steps of this approach are same as that of first approach. Both approaches only differ in terms of input given to the RNN.

The dataset used for both the approaches consists of Argentinean Sign Language (LSA) [14] gestures, with around 2300 videos for 46 gestures categories as shown in Table 1. Ten non-expert subjects executed the 5 repetitions of each gesture, i.e., 50 videos per gesture. Out of these 50 video samples, 40 videos per gesture were used for training, and 10 videos were used for testing. Thus, the training dataset had 1840 videos and the test dataset had 460 videos.

Table 1 Gestures used for training/testing

ID	Name	ID	Name	ID	Name
1	Son	17	Where	33	Barbeque
2	Food	18	Breakfast	34	Spaghetti
3	Trap	19	Catch	35	Patience
4	Accept	20	Name	36	Rice
5	Opaque	21	Yogurt	37	To-Land
6	Water	22	Man	38	Yellow
7	Colors	23	Drawer	39	Give
8	Perfume	24	Bathe	40	Away
9	Born	25	Country	41	Copy
10	Help	26	Red	42	Skimmer
11	None	27	Call	43	Sweet milk
12	Deaf	28	Run	44	Chewing gum
13	Enemy	29	Bitter	45	Photo
14	Dance	30	Map	46	Thanks
15	Green	31	Milk		
16	Coin	32	Uruguay		

4 Result

The prediction approach got an accuracy of 80.87% by recognizing 370 gestures correctly a test set of 460, while the pool layer approach scored 95.21% by recognizing 438 gestures.

It may be observed from Table 2 that all test videos for gesture “son” in prediction approach were incorrectly classified as they were misclassified as “colors”.

It may be observed from Fig. 6a, b that the gestures, “Son” and “Colors”, involve moving hands horizontally toward the right. The only difference is that in case of symbol “Son” it is done by holding two fingers up, while in case of “Colors” it is done with only one finger up. This extremely high level of similarity may be the reason behind misclassification of all “Son” gesture as “Color”.

Table 2 Accuracy (in percent) per gesture using Approach 1 and Approach 2

ID	Gesture	App_1	App_2	ID	Gesture	App_1	App_2
1	Name	100	100	24	Spaghetti	70	100
2	Yogurt	100	100	25	Patience	70	100
3	Accept	30	90	26	Deaf	80	90
4	Man	70	100	27	Enemy	50	90
5	Drawer	100	100	28	Dance	100	90
6	Bathe	100	100	29	Rice	100	100
7	Opaque	70	90	30	To-Land	50	100
8	Country	100	100	31	Yellow	100	100
9	Water	60	90	32	Green	80	90
10	Red	100	100	33	Give	100	100
11	Call	90	100	34	Food	50	80
12	Colors	90	90	35	Away	80	100
13	Run	100	100	36	Copy	80	100
14	Bitter	100	100	37	Coin	100	90
15	Perfume	60	90	38	Where	100	90
16	Map	100	100	39	Skimmer	80	100
17	Born	80	90	40	Trap	100	80
18	Help	70	90	41	Sweet milk	100	100
19	Milk	100	100	42	Breakfast	100	90
20	None	80	90	43	Chewing gum	100	100
21	Uruguay	100	100	44	Photo	90	100
22	Son	0	80	45	Thanks	30	100
23	Barbeque	70	100	46	Catch	40	90

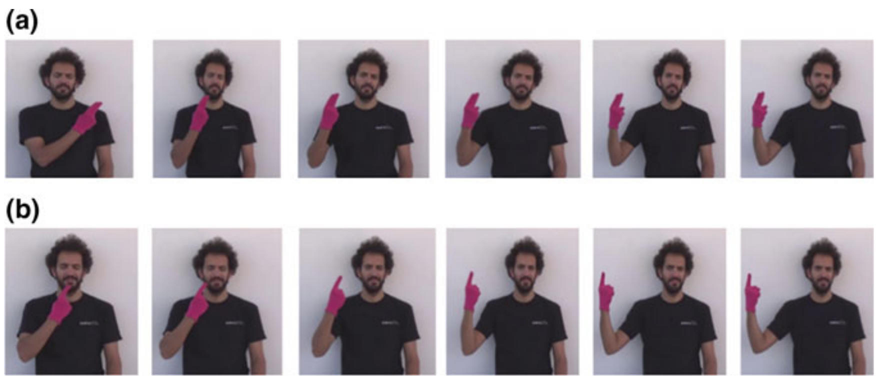

Fig. 6 **a** Set of few extracted frames of gesture "Son" [14]. **b** Set of few extracted frames of gesture "Color" [14]

4.1 *Comparison Between the Two Approaches*

The pool layer approach achieves a better accuracy than the prediction approach due to the increased size of the feature vector per frame being given to the RNN. In the prediction approach, for each frame the prediction of the CNN was given as input to the RNN. The prediction by the CNN was a list of values where the *i*th value denotes the probability of the frame belonging to the *i*th category. Hence, the size of the feature vector per frame was 46 in this work.

In the pool layer approach, for each frame the output of the pool layer, before it is made into a prediction, was given as input. The pool layer gives a 2048-dimensional vector that represents the convoluted features of the image. If each category is observed individually, it can be seen that for most of the categories the pool layer approach proves to be better. Though the high number of features have improved the correct classification rate for most of the gestures, but might have caused the RNN to discover random noise in the finite training set. Hence, it learnt some features that did not add any value to the judgement but may have led to overfitting.

5 Conclusion

In this work, we presented a vision-based system to interpret isolated hand gestures from the Argentinean Sign Language. This work used two different approaches to classify on the spatial and temporal features. CNN was used to classify on the spatial features, whereas RNN was used to classify on the temporal features. We obtained an accuracy of 95.217%. This shows that CNN along with RNN can be successfully used to learn spatial and temporal features and classify sign language gesture videos.

References

1. Ronchetti, F., Quiroga, F., Estrebou, C.A., Lanzarini, L.C.: Handshape recognition for Argentinian sign language using probsom. J. Comput. Sci. Technol. **16** (2016)
2. Singha, J., Das, K.: Automatic Indian Sign Language recognition for continuous video sequence. ADBU J. Eng. Technol. **2**(1) (2015)
3. Tripathi, K., Nandi, N.B.G.C.: Continuous Indian Sign Language gesture recognition and sentence formation. Procedia Comput. Sci. **54**, 523–531 (2015)
4. Nandy, A., Prasad, J.S., Mondal, S., Chakraborty, P., Nandi, G.C.: Recognition of isolated Indian Sign Language gesture in real time. Inf. Process. Manag., 102–107 (2010)
5. Pigou, L., Dieleman, S., Kindermans, P.-J., Schrauwen, B.: Sign language recognition using convolutional neural networks. In: Workshop at the European Conference on Computer Vision 2014, pp. 572–578. Springer International Publishing (2014)
6. Sharma, R., Bhateja, V., Satapathy, S.C., Gupta, S.: Communication device for differently abled people: a prototype model. In: Proceedings of the International Conference on Data Engineering and Communication Technology, pp. 565–575. Springer, Singapore (2017)
7. Masood, S., Thuwal, H.C., Srivastava, A.: American sign language character recognition using convolution neural network. In: Proceedings of Smart Computing and Informatics, pp. 403–412. Springer, Singapore (2018)
8. Vicars, W.: Sign language resources at LifePrint.com. http://www.lifeprint.com/asl101/pages-signs/f/friend.htm. Accessed 23 Sept 2017
9. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
10. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., et al.: Tensorflow: large-scale machine learning on heterogeneous distributed systems (2016). arXiv preprint arXiv:1603.04467
11. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
13. Kingma, D., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint arXiv:1412.6980
14. Ronchetti, F., Quiroga, F., Estrebou, C.A., Lanzarini, L.C., Rosete, A.: LSA64: an Argentinian sign language dataset. In: XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016) (2016)