Computer Assisted Methods in Engineering and Science
30(3): 263–276, 2023, doi: 10.24423/cames.559

This article belongs to the *Special Issue on Digital Twin Empowered Internet of Intelligent Things for Engineering Cyber-Physical Human Systems* edited by C. Venkatesan, Y.-D. Zhang, Q. Xin

A Deep Hybrid Model for Human-Computer Interaction Using Dynamic Hand Gesture Recognition

Brindha RAMALINGAM1)*, Geetha ANGAPPAN2)

Computer Science and Engineering, Annamalai University, Annamalai Nagar, Chidambaram, India; e-mail: aucsegeetha@yahoo.com

* *Corresponding Author e-mail: brinzram@gmail.com*

Dynamic hand gestures attract great interest and are utilized in different fields. Among these, man-machine interaction is an interesting area that makes use of the hand to provide a natural way of interaction between them. A dynamic hand gesture recognition system is proposed in this paper, which helps to perform control operations in applications such as music players, video games, etc. The key motivation of this research is to provide a simple, touch-free system for effortless and faster human-computer interaction (HCI). As this proposed model employs dynamic hand gestures, HCI is achieved by building a model with a convolutional neural network (CNN) and long short-term memory (LSTM) networks. CNN helps in extracting important features from the images and LSTM helps to extract the motion information between the frames. Various models are constructed by differing the LSTM and CNN layers. The proposed system is tested on an existing EgoGesture dataset that has several classes of gestures from which the dynamic gestures are utilized. This dataset is used as it has more data with a complex background, actions performed with varying speeds, lighting conditions, etc. This proposed hand gesture recognition system attained an accuracy of 93%, which is better than other existing systems subject to certain limitations.

Keywords: dynamic hand gesture, human-computer interaction, long short-term memory, convolutional neural network.

1. Introduction

Human-computer interaction (HCI) provides continuous developments to make our daily life easier, more natural, and less costly. Devices such as keyboards, mouse, etc., are mainly used to interact with computers [1]. Compared to these devices, interaction using gestures is the more convenient, practical, and natural way. Gestures are communicative, suggestive motions articulated through physical actions of many body parts such as the head, hands, fingers, etc., that convey proper information [2]. Among these gestures, HCI through hand is an attractive method and also helps to communicate more data in minimal time. These gestures are a standard type of body language for conveying defiance and reactions. Hand gestures are classified into static and dynamic gestures [3]. There are two methods used to interpret hand gestures for HCI: the data gloves-based method and the vision-based method [4]. The data gloves-based method needs an additional device which in turn requires extra cables, whereas the vision-based method needs only a camera or a sensor to capture images or videos.

The objective of this work is to build a touch-free, easy-to-use system that can be very helpful in pandemic situations such as COVID-19. Thus, a vision-based dynamic hand gesture recognition system is proposed in this paper. It involves a few dynamic hand gestures that are classified using CNN and LSTM deep neural network (DNN) models. CNN is chosen for feature extraction purposes due to its capability to classify, segment, and retrieve images [5]. LSTM is chosen for sequence-to-sequence recognition as it performs well in remembering long-term dependencies. Dynamic hand gestures will be very helpful in many applications such television control systems, robot control, smart home appliances (e.g., control of lamp), media player, recommendation system, MS office, document reader, etc., for man-machine interaction [6].

The remaining portion of the paper is organized as follows: Sec. 2 describes related work, Sec. 3 presents details about the input data and the classifiers, Sec. 4 provides a brief explanation of the proposed architecture, Sec. 5 elaborates on experimentations and architecture details, Sec. 6 provides results and discussions, and Sec. 7 presents conclusion and directions for future work.

2. Related work

To achieve human-computer interaction, the authors in [7, 8] developed various models based on static and dynamic hand gestures. The authors in [1] consider static hand gestures to show the finger count. The authors use AdaBoost as this method is well suited for this work. In [9], dynamic hand gestures are used and a recognition system is built using a hierarchical architecture. The work is tested using NVIDIA Dynamic Hand Gesture and EgoGesture datasets attaining accuracy of 83.82% and 94.04%, respectively. In addition, resources are efficiently used, activations are made single-time and early detections are conducted. Static hand gestures are used in [19] to operate PowerPoint presentations, recommendation systems, etc. Fingertip detection is achieved in [5] and also hand gestures are recognized using the convex hull technique and region growing segmentation. respectively. For faster HCI, single-handed thumb-to-finger micro gestures are used for recognition with CNN and depth-sensing networks. It achieved 91% accuracy for 8 gesture classes. In [2], for hand videos, keyframes are extracted and feature fusion is performed to improve recognition accuracy. The paper [15] utilized head gestures and hand gestures for achieving HCI. Hand gestures are recognized by segmenting them initially from video, and hand shape and motion are calculated. In [12] and [13], fusion of CNN and LSTM models are performed to learn spatial and temporal features. A hybrid model is built in [14] to check for humans wearing masks where deep learning is used for feature extraction and machine learning for classification.

Head gestures are recognized using an optical flow technique and finite-state automata. These gestures help in controlling computers and other applications. The work in [20] proposed a deep model for video classification. It includes two separate streams (spatial and temporal) for recognition using convnets. This system is evaluated using HMDB and UCF datasets. In [15], a hand gesture recognition system is developed for HCI. A metric for distance and an algorithm for recognition are proposed and demonstrated using their self-dataset and two existing datasets. Thus, this system attains high performance and recognition speed is fast Studies on segmentation techniques are provided in [16], where they are very useful in the detection/recognition of face and hand, image analysis in the medical field, etc. An algorithm is presented in [10] that uses optimization strategies to increase robustness and estimation speed by using randomized decision forests in hand gesture recognition. This model utilizes fewer computational resources. A review of the CNN model is given in [17], which provides more information on feature layers and CNN operations. Human emotions are recognized using LSTM-RNN network in [21] where prediction is done using machine learning models.

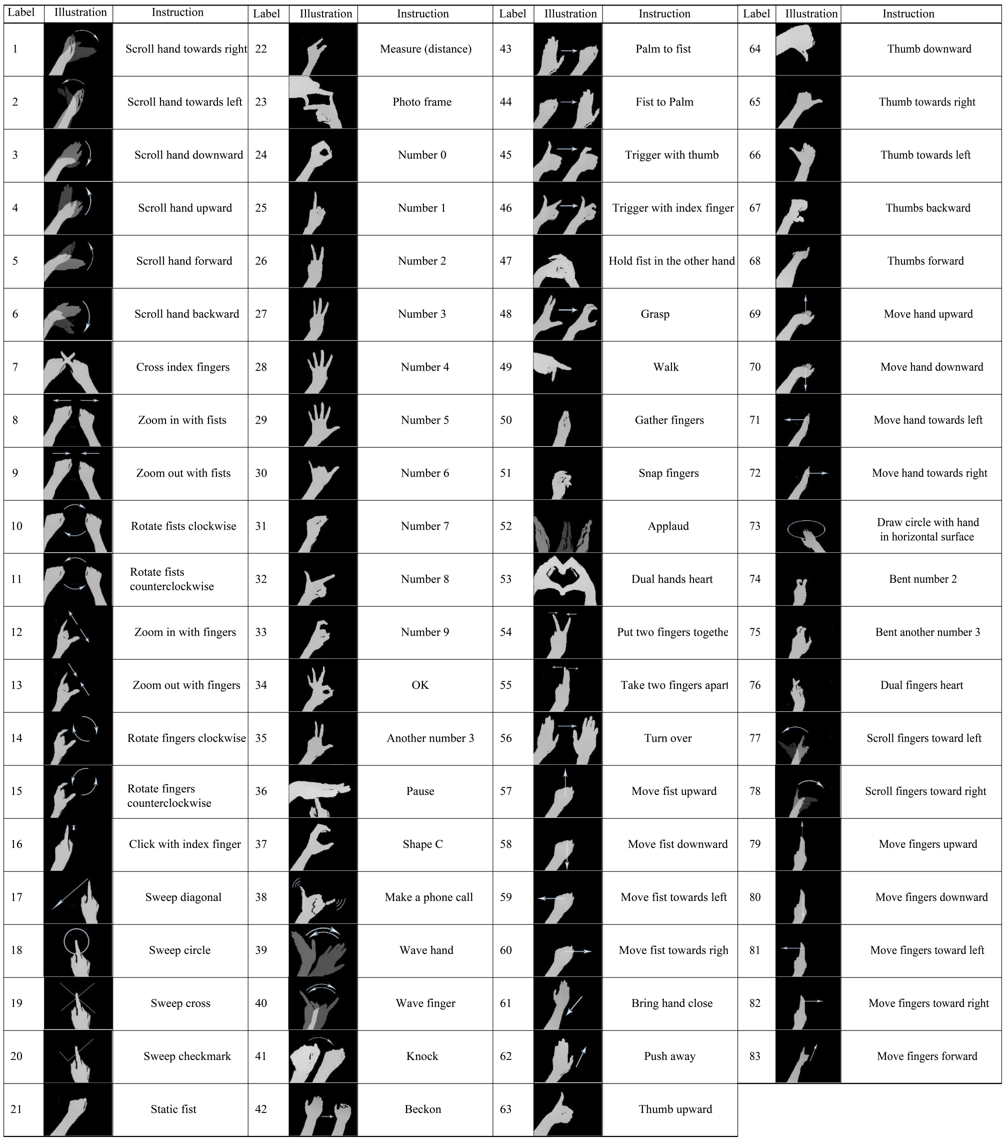
A hand gesture recognition system is developed in [22] for Indian Sign Language (ISL) for static and dynamic types. In this work, dynamic time warping is performed to eliminate recurrent frames, optimal features are selected and recognition is conducted through a neural network. The work proposed in [23] also uses hand gestures to build a lightweight model in order to help people with certain disabilities to communicate with other people. YOLO (You Look Only Once) V3 and DarkNet-53 CNN are used for recognizing gestures. The model is built for both static and dynamic gestures, achieving an accuracy of 97.68%. The paper [24] presented a hand pose-based gesture recognition system using CNN and LSTM. This work is very useful when gestures are with comparable motions or temporal patterns. In [25], a gesture recognition model is built for extracting spatio-temporal information efficiently. This work utilizes the ResC3D network and convolutional LSTM to capture the information in every stage of feature learning.

3. Materials and methods

This section briefly discussed the dataset used in this work and the deep networks utilized for dynamic hand gesture recognition.

3.1. Dataset

In this work, the EgoGesture dataset [11, 18] is used which is a large dataset for both static and dynamic gesture recognition. This dataset contains static and dynamic gestures of 83 classes, as shown in Fig. 1. The videos of this dataset were taken using 50 subjects under 6 different indoor and outdoor scenes with varying timing, lighting conditions, and backgrounds. So, there were 300 videos with all the classes in RGB and depth format.



Label	Illustration	Instruction	Label	Illustration	Instruction	Label	Illustration	Instruction	Label	Illustration	Instruction
1		Scroll hand towards right	22		Measure (distance)	43		Palm to fist	64		Thumb downward
2		Scroll hand towards left	23		Photo frame	44		Fist to Palm	65		Thumb towards right
3		Scroll hand downward	24		Number 0	45		Trigger with thumb	66		Thumb towards left
4		Scroll hand upward	25		Number 1	46		Trigger with index finger	67		Thumbs backward
5		Scroll hand forward	26		Number 2	47		Hold fist in the other hand	68		Thumbs forward
6		Scroll hand backward	27		Number 3	48		Grasp	69		Move hand upward
7		Cross index fingers	28		Number 4	49		Walk	70		Move hand downward
8		Zoom in with fists	29		Number 5	50		Gather fingers	71		Move hand towards left
9		Zoom out with fists	30		Number 6	51		Snap fingers	72		Move hand towards right
10		Rotate fists clockwise	31		Number 7	52		Applaud	73		Draw circle with hand in horizontal surface
11		Rotate fists counterclockwise	32		Number 8	53		Dual hands heart	74		Bent number 2
12		Zoom in with fingers	33		Number 9	54		Put two fingers togethe	75		Bent another number 3
13		Zoom out with fingers	34		OK	55		Take two fingers apart	76		Dual fingers heart
14		Rotate fingers clockwise	35		Another number 3	56		Turn over	77		Scroll fingers toward left
15		Rotate fingers counterclockwise	36		Pause	57		Move fist upward	78		Scroll fingers toward right
16		Click with index finger	37		Shape C	58		Move fist downward	79		Move fingers upward
17		Sweep diagonal	38		Make a phone call	59		Move fist towards left	80		Move fingers downward
18		Sweep circle	39		Wave hand	60		Move fist towards righ	81		Move fingers toward left
19		Sweep cross	40		Wave finger	61		Bring hand close	82		Move fingers toward right
20		Sweep checkmark	41		Knock	62		Push away	83		Move fingers forward
21		Static fist	42		Beckon	63		Thumb upward			

FIG. 1. 83 classes of EgoGesture dataset [26].

The dataset of this work comprises particular videos chosen from this EgoGesture dataset, arranged into four categories (scroll left to right, scroll right to left, zoom in, zoom out), shown in Fig. 2. These four classes of videos are extracted manually using the filmora application by examining each subject's videos. Each subject performs dynamic gestures in a different order and at different speeds. Thus, this dataset is used to construct a dynamic gesture recognition system.

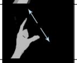
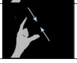
Label	Illustration	Instruction	Label
1		Scroll hand towards right	22
2		Scroll hand towards left	23
12		Zoom in with fingers	33
13		Zoom out with fingers	34

FIG. 2. Four hand actions used for our work.

3.2. Methods

A brief explanation of two models, CNN and LSTM, are given in this section.

3.2.1. Convolutional Neural Network (CNN). CNN is a type of deep neural network (DNN) that is primarily used in the field of image processing. It comprises neurons capable of self-optimizing over learning. These neurons are associated with three dimensions: height, width, and depth. The pixel values of the image are in the input layer. Other than the input layer, there are three important layers in CNN. *Convolutional layers* apply filters to the images. Then, in the *pooling layer*, parameters are reduced and computations are minimized. The pooling layer output is compressed and fed to *an entirely connected layer*. The last layer of the network is the output layer, where the classification of classes is performed.

3.2.2. Long Short-Term Memory (LSTM). LSTM, a gentle recurrent neural network (RNN), is evolved by overcoming short-term dependencies and vanishing gradient problems. LSTM is capable of learning long-term dependencies. The major part of LSTM is the cell state (memory of LSTM) and three gates (helps in adding or removing information to/from the cell state). A cell state transfers information right down the sequence chain. The input gate chooses what novel information should stay in the cell state. The forget gate helps to throw out unnecessary information. Activation is provided through the output gate to the final output of the LSTM block. As this research work is based on end-to-end sequence recognition, such a model is employed, which is very appropriate.

4. Proposed architecture

This section presents the architecture of the proposed system built on the CNN-LSTM model (Fig. 3) and a brief explanation of the models used to build it.

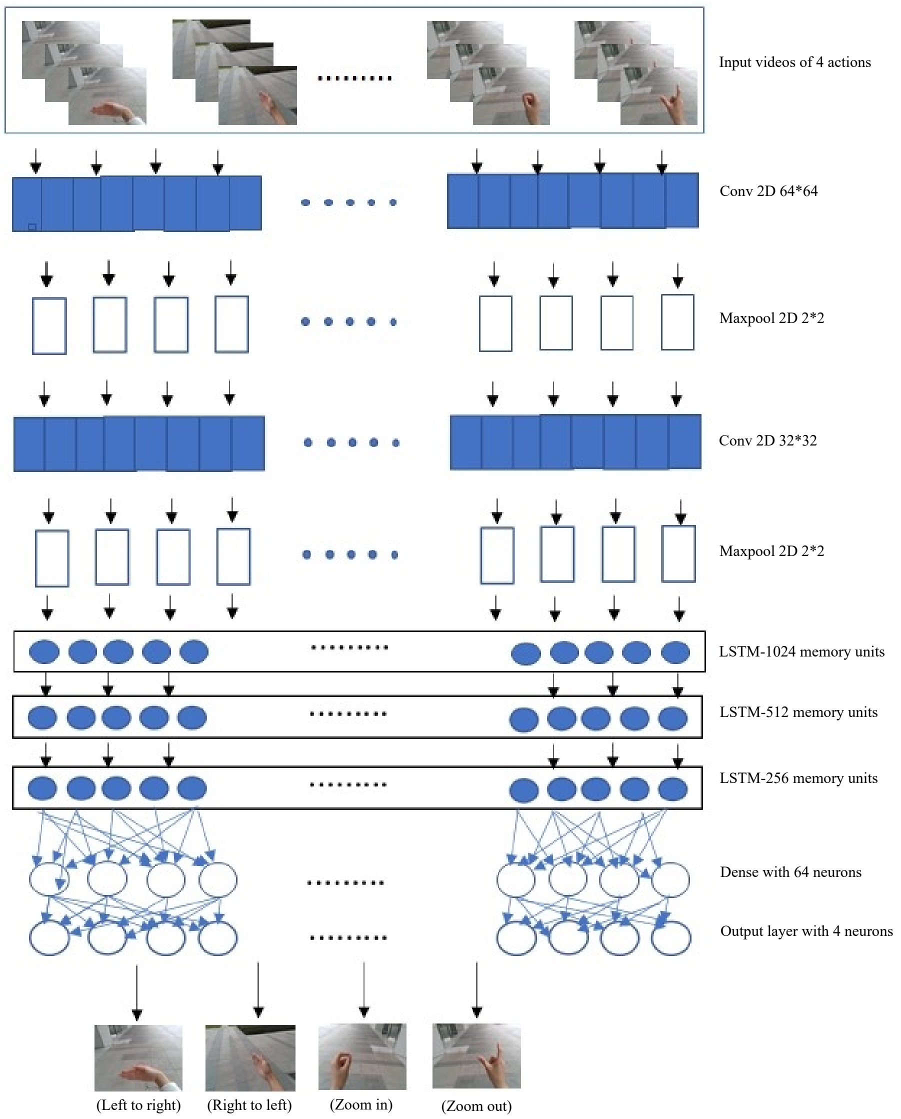


Fig. 3. Dynamic hand gesture recognition system.

A video is a collection of images in a specific order. Each video has both spatial and temporal information in individual images and the order in which they are arranged. In this work, a hybrid architecture is built to model these two pieces of information. The architecture uses both a CNN and an RNN. LSTM is the RNN used in this work. CNN comprises convolutional layers that extract spatial information. RNN comprises recurrent layers that extract temporal information. This hybrid network is built with 2D (two-dimensional) convolutional layers trailed by 2D max-pooling layers. The pooling layer output is given as input to the next layer, an LSTM layer.

The LSTM layer output is given as input to the dense layer. The last dense layer represents the output which is nothing but four actions. This network is built with a various number of convolutional layers, LSTM layers, and dense layers. The steps involved in a dynamic hand gesture recognition system are as follows. Videos of four actions are chosen, which show only the hand region above the wrist, as shown in Fig. 4. The extracted video portions are approximately less or equal to 6 seconds. Frames are extracted from each video, which is in RGB format, as shown in Fig. 4. Extracted frames are resized to a resolution of 480×640 pixels. Then, zero padding (given in Sec. 5) is performed to make an equal number of frames for all the videos. Zero-padded reshaped frames are fed as input. Also, class names, in the form of strings, are translated into integer values before they are passed to the classifier. Then, hand features are extracted using the CNN model, and dynamic gestures are classified using the LSTM model.

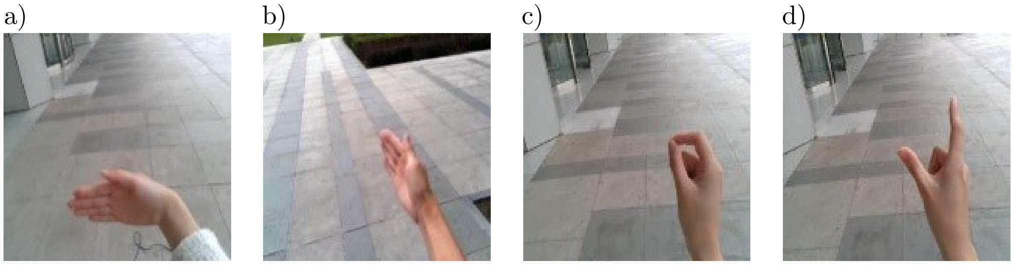


Fig. 4. Sample actions used for our work:
a) right to left, b) left to right, c) zoom in, d) zoom out.

5. Experimentations and architecture details

The experiments are conducted using Python 3.1 on a processor of Intel Core i5-8300H CPU @ 2.30GHz with 8.00 GB RAM, and the platform is Windows 10. The system is tested using videos of four actions from the EgoGesture dataset. Input is given as videos of four classes. The total number of videos used for training is 80 (20 videos for each class), and for testing it is 40 (10 videos for each class). Each video is less or equal to 6 seconds. For each video, the extraction of frames is 16 frames per second (fps). Zero Padding: as videos are of different lengths, zero padding is performed to make an equal number of frames for each video. To perform zero-padding, a maximum number of frames is set to a particular value. This value is chosen based on the video, which has a maximum of seconds. In this dataset, the longest video is 6 seconds. As we are extracting 16 fps, the longest video contains 96 frames. Thus, the maximum number of frames is set to 96. The videos, less than 6 seconds, are padded with zeros until the frame length is equal to 96. Thus, all the videos are of the same length after zero padding, which is easy for classification.

The system is initially built with only the LSTM model, as our research problem is based on sequence-to-sequence classification. The LSTM model is capable of remembering long-term dependencies. The architecture details of the LSTM model developed are given in Subsec. 5.1. But it needs a greater number of parameters, which results in time-consuming training. So, the LSTM model is combined with the CNN model, which forms a hybrid CNN-LSTM network. This CNN-LSTM network is built with convolutional layers, pooling layers, and the last dense layer of the CNN model is removed. Instead of that, one or more LSTM layers are added. This hybrid network helps in extracting more features with lesser training time without affecting the performance of the system.

5.1. LSTM model architecture

The LSTM model input are the video frames that are resized to 112×112. This model is constructed with two hidden LSTM layers and a dense layers. Memory units in LSTM layers are 512 and 256. Rectified Linear Unit (ReLu) is the activation function used in all the layers where the output dense layer contains softMax activation. Hyperparameters utilized in this model are as follows: 50 epochs, the learning rate is 0.001, the optimizer is Adam, the accuracy metric is "accuracy", and loss metric is "categorical_crossentropy". The output layer consists of four neurons that represent the four-hand actions.

5.2. CNN-LSTM model architecture

In the CNN-LSTM network, multiple models are built manually by varying the layers in the CNN and LSTM models, and a suitable model for this proposed work is found. Among those multiple models, three models with better performance are explained in this section. Model 'a' is built with two CNN layers and one LSTM layer (vanilla LSTM). Model 'b' has two CNN layers and two LSTM layers. Model 'c' has two CNN layers and three LSTM layers. Only the architecture details of model 'c' are given below because it achieves high performance compared to the other two models. Also, the CNN-LSTM network performance did not increase beyond these number of layers for our proposed work.

Model 'c' is initially built with two 1D convolutional layers and 1D max-pooling layers. The filters used in convolutional layers are 64 and 32. The kernel size is chosen to be 3 in both convolutional layers. In max-pooling layers, the pool size is chosen to be 2. The pooling layer output is fed as input to the LSTM layer. Three LSTM layers are used in which memory units in each layer are 1024, 512, and 256, and the softMax activation function is used as well. Then a dropout layer of 0.2 is used before the dense layer. Two dense layers are used, one with 64 neurons and softMax activation function, and the other is the output layer with four classes representing left to right, right to left, zoom in, and zoom out actions. The model is compiled with the following hyperparameters: loss is sparse categorical cross-entropy, accuracy metrics is categorical accuracy, 30 epochs, and Adam optimizer where the default learning rate is 0.001.

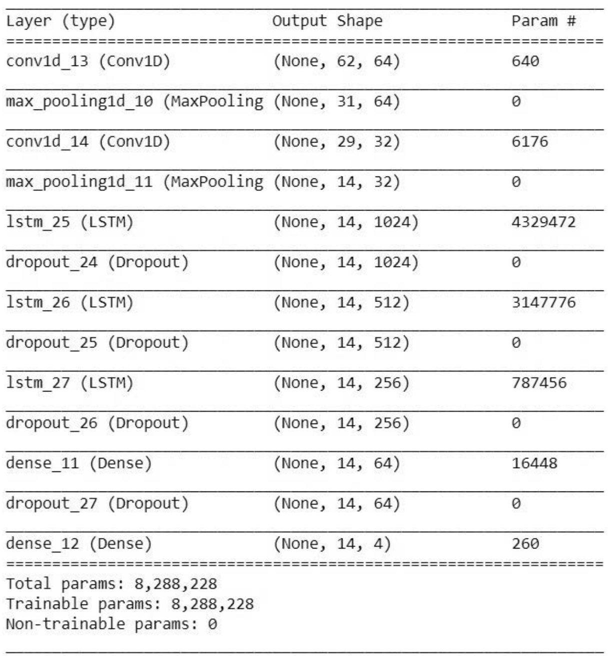
6. Results and discussion

In this work, dynamic hand gesture recognition system performance is calculated using accuracy metrics. Accuracy is a metric used to analyze whether a model yields better performance or not. The accuracy of LSTM and CNN-LSTM models is shown in Table 1. The summary of CNN-LSTM model 'c' with the highest accuracy for our proposed system is shown in Fig. 5. The trainable parameters in Fig. 5 are the features extracted.

Table 1. Accuracy of LSTM and CNN-LSTM models.

S. No.	Name of the model		No. of layers	Accuracy [%]
1.	LSTM		1	50.25
			2	63.78
			3	75.00
2.	CNN-LSTM	'a'	2 CNN – 1 LSTM	78.89
		'b'	2 CNN – 2 LSTM	81.34
		'c'	2 CNN – 3 LSTM	93.00

Figure 6 shows the CNN-LSTM model 'c' training and validation accuracy values during each epoch. Figure 7 shows training and validation loss values obtained during each epoch. In Fig. 6, it is observed that training and validation accuracy gradually increases to 30 epochs. When the model runs beyond 30 epochs, accuracy starts to decrease and thus the model becomes overfitting. Thus, for our proposed work with limited data, epochs are fixed as 30. Figure 7 shows the loss values of CNN-LSTM model 'c' during each epoch. It is observed that loss values start decreasing gradually in the first 5 epochs and rapidly decrease from the 5th epoch to the 14th epoch. After the 14th epoch, it is a slow decrease in loss till the 25th epoch. Then, there is no decrease in loss till the last epoch value.



```
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_13 (Conv1D)           (None, 62, 64)            640
_________________________________________________________________
max_pooling1d_10 (MaxPooling (None, 31, 64)            0
_________________________________________________________________
conv1d_14 (Conv1D)           (None, 29, 32)            6176
_________________________________________________________________
max_pooling1d_11 (MaxPooling (None, 14, 32)            0
_________________________________________________________________
lstm_25 (LSTM)               (None, 14, 1024)          4329472
_________________________________________________________________
dropout_24 (Dropout)         (None, 14, 1024)          0
_________________________________________________________________
lstm_26 (LSTM)               (None, 14, 512)           3147776
_________________________________________________________________
dropout_25 (Dropout)         (None, 14, 512)           0
_________________________________________________________________
lstm_27 (LSTM)               (None, 14, 256)           787456
_________________________________________________________________
dropout_26 (Dropout)         (None, 14, 256)           0
_________________________________________________________________
dense_11 (Dense)             (None, 14, 64)            16448
_________________________________________________________________
dropout_27 (Dropout)         (None, 14, 64)            0
_________________________________________________________________
dense_12 (Dense)             (None, 14, 4)             260
=================================================================
Total params: 8,288,228
Trainable params: 8,288,228
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5. Summary of CNN-LSTM model ‘c’ architecture.

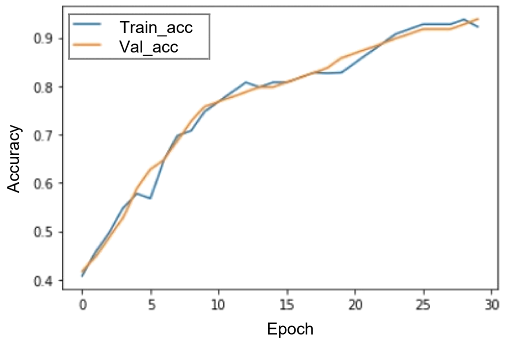


Fig. 6. CNN-LSTM model ’c’ accuracy.

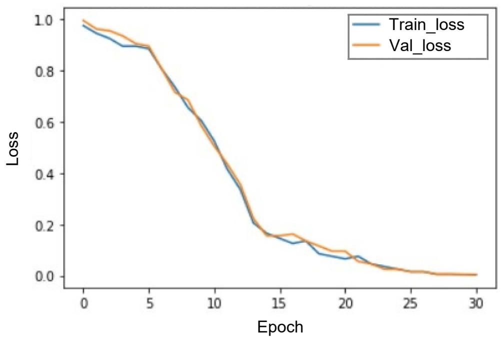


Fig. 7. CNN-LSTM model ’c’ loss.

Table 1 displays the accuracy values of the LSTM model and the combined CNN-LSTM models constructed using a different number of LSTM and CNN layers. Initially, LSTM models are constructed with 1, 2, and 3 LSTM layers, respectively. All these three models' accuracy is less than or equal to 75%. To achieve better performance, we tried combining CNN and LSTM layers where CNN layers are fixed as two and LSTM layers are varied. Thus, three CNN-LSTM models are built where the performance started improving. CNN-LSTM model 'a' has 78.89%, model 'b' has 81.34, and model 'c' has 93% accuracy. So, by comparing with other models, it is observed that CNN-LSTM with two CNN layers and three LSTM layers, i.e., the CNN-LSTM model 'c' performed well by achieving an accuracy of 93%.

The proposed system for dynamic hand gesture recognition was executed with the LSTM model and hybrid CNN-LSTM model. In the LSTM model, many models were built with one LSTM layer (vanilla LSTM), two LSTM layers, and three LSTM layers (stacked LSTM) separately. More layers were tried where the accuracy did not increase. Among these models, LSTM with three layers produced an accuracy of 75%. Then, the CNN-LSTM model was executed with a varying number of convolutional layers (1, 2, 3) and LSTM layers (1, 2, 3). Hyperparameter selection was made based on the trial-and-error method. The model with two convolutional layers and three LSTM layers yielded 93% accuracy.

The proposed dynamic hand gesture recognition system with the CNN-LSTM model works well and provides good performance compared to existing models. The major advantage of this hybrid model is that more features can be extracted requiring less training time and performance will not be affected. Also, the EgoGesture dataset utilized in this work is a large-scale dataset prepared under different circumstances. Thus, this dataset makes this proposed system robust and helps to work in any situation.

7. Conclusion and future work

This paper provided a HCI system through dynamic hand gestures, which are very helpful and safe during COVID-19 pandemic situations. This proposed system comprises four actions: left-to-right, right-to-left, zoom-in, and zoom-out, performed without the use of a keyboard, mouse, or any other device but only a hand. This system is robust because it was tested on the EgoGesture dataset, which consists of actions performed with different timing, varying lighting conditions, and complex backgrounds. As this work is a sequence-based problem, the CNN-LSTM model is employed in the process of extracting features and recognizing the actions. The system achieved an accuracy of 93%. Table 1 shows that the hybrid CNN-LSTM model 'c' worked well compared to the plain LSTM model. The output of this proposed work can be used for hand-free operations in mobile interfacing, media interfacing, etc., which will be helpful in this fast-moving world. For example, in mobile, hand-free operations such as scrolling windows, zooming in and out, volume increase and decrease, etc., operations can be performed in an easy and quick way.

In the proposed system, gestures were sometimes not recognized properly when lighting conditions were bad. The system was tested only on the existing dataset and did not perform in real time. Only four classes were considered in our work. More classes such as scroll up, scroll down, play, pause, etc. can be included. Future work can be conducted by considering these drawbacks and a more robust, complete, user-friendly system can be built.

Acknowledgements

We are thankful to the National Laboratory of Pattern Recognition, Institute of Automation, the Chinese Academy of Sciences for providing us with the EgoGesture dataset.

References

1. R.M. Gurav, P.K. Kadbe, Real time finger tracking and contour detection for gesture recognition using OpenCV, [in:] *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 974–977, IEEE, 2015, doi: 10.1109/IIC.2015.7150886.
2. H. Tang, H. Liu, W. Xiao, N. Sebe, Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion, *Neurocomputing*, **331**(C): 424–433, 2019, doi: 10.1016/j.neucom.2018.11.038.
3. N.A. Ibraheem, R.Z. Khan, M.M. Hasan, Comparative study of skin color-based segmentation techniques, *International Journal of Applied Information Systems (IJAIS)*, **5**(10): 24–34, 2013, doi: 10.5120/ijais13-450985.
4. M. Alhussein, K. Aurangzeb, S.I. Haider, Hybrid CNN-LSTM model for short-term individual household load forecasting, *IEEE Access*, **8**: 180544–180557, 2020, doi: 10.1109/ACCESS.2020.3028281.
5. R.M. Prakash, T. Deepa, T. Gunasundari, N. Kasthuri, Gesture recognition and finger-tip detection for human computer interaction, [in:] *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, pp. 1–4, IEEE, 2017, doi: 10.1109/ICIIECS.2017.8276056.
6. M. Soliman, F. Mueller, L. Hegemann, J.S. Roo, C. Theobalt, J. Steimle, Finger input: Capturing expressive single-hand thumb-to-finger microgestures, [in:] *2018 International Conference on Interactive Surfaces and Spaces (ICISS)*, pp. 177–187, ACM, 2018, doi: 10.1145/3279778.3279799.
7. F. Chen, J. Deng, Z. Pang, M. Baghaei Nejad, H. Yang, G. Yang, Finger angle-based hand gesture recognition for smart infrastructure using wearable wrist-worn camera, *Applied Sciences*, **8**(3): 369, 2018, doi: 10.3390/app8030369.

8. N.L. Hakim, T.K. Shih, S.P.K. Arachchi, W. Aditya, Y.C. Chen, C.Y. Lin, Dynamic hand gesture recognition using 3DCNN and LSTM with FSM context-aware model, *Sensors*, **19**(24): 5429, 2019, doi: 10.3390/s19245429.

9. O. Kopuklu, A. Gunduz, N. Kose, G. Rigoll, Real-time hand gesture detection and classification using convolutional neural networks, [in:] *2019 International Conference on Automatic Face & Gesture Recognition (ICAFGR)*, pp. 1–8, IEEE, 2019, doi: 10.48550/ARXIV.1901.10323.

10. S. Sridhar, F. Mueller, A. Oulasvirta, C. Theobalt, Fast and robust hand tracking using detection-guided optimization, [in] *2015 Conference on Computer Vision and Pattern Recognition*, pp. 3213–3221, IEEE, 2015, doi: 10.1109/cvpr.2015.7298941.

11. C. Cao, Y. Zhang, Y. Wu, H. Lu, J. Cheng, Egocentric gesture recognition using recurrent 3d convolutional neural networks with spatiotemporal transformer modules, [in:] *2017 International conference on computer vision (ICCV)*, pp. 3763–3771, IEEE, 2017, doi: 10.1109/ICCV.2017.406.

12. H. Gammulle, S. Denman, S. Sridharan, C. Fookes, Two stream LSTM: A deep fusion framework for human action recognition, [in:] *2017 Winter Conference on Applications of Computer Vision (WACV)*, pp. 177–186, IEEE, 2017, doi: 10.1109/WACV.2017.27.

13. M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, A. Baskurt, Sequential deep learning for human action recognition, [in:] *2011 International workshop on human behavior understanding*, pp. 29–39, Springer, 2011, doi: 10.1007/978-3-642-25446-8_4.

14. M. Loey, G. Manogaran, M.H.N. Taha, N.E.M. Khalifa, A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic, *Measurement*, **167**: 108288, 2021, doi: 10.1016/j.measurement.2020.108288.

15. A. Agrawal, R. Raj, S. Porwal, Vision-based multimodal human-computer interaction using hand and head gestures, [in:] *2013 Conference on Information & Communication Technologies*, pp. 1288–1292, IEEE, 2013, doi: 10.1109/CICT.2013.6558300.

16. C. Wang, Z. Liu, S.C. Chan, Super pixel-based hand gesture recognition with kinect depth camera, *IEEE transactions on multimedia*, **17**(1): 29–39, 2014, doi: 10.1109/TMM.2014.2374357.

17. M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, [in:]*2014 European Conference on computer vision (ECCV)*, **8689**: 818–833, Springer, 2014, doi: 10.1007/978-3-319-10590-1_53.

18. Y. Zhang, C. Cao, J. Cheng, H. Lu, EgoGesture: A new dataset and benchmark for egocentric hand gesture recognition, *IEEE Transactions on Multimedia*, **20**(5): 1038–1050, 2018, doi: 10.1109/TMM.2018.2808769.

19. R.P. Sharma, G.K. Verma, Human computer interaction using hand gesture, *Procedia Computer Science*, **54**: 721–727, 2015, doi: 10.1016/j.procs.2015.06.085.

20. K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, [in:] *2014 International Conference on Neural Information Processing Systems*, Vol. 1, pp. 568–576, 2014, doi: 10.48550/ARXIV.1406.2199.

21. L. Chao, J. Tao, M. Yang, Y. Li, Z. Wen, Long short-term memory recurrent neural network based encoding method for emotion recognition in video, [in:] *2016 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2752–2756, IEEE, 2016, doi: 10.1109/ICASSP.2016.7472178.

22. K. Manisha, K. Artik, Automatic hand gesture recognition using hybrid meta-heuristic-based feature selection and classification with dynamic time warping, *Computer Science Review*, **39**: 100320, 2021, doi: 10.1016/j.cosrev.2020.100320.

23. A. Mujahid *et al.*, Real-time hand gesture recognition based on deep learning YOLOv3 model, *Applied Sciences*, **11**(9): 4164, 2021, doi: 10.3390/app11094164.

24. C. Li, S. Li, Y. Gao, X. Zhang, W. Li, A two-stream neural network for pose-based hand gesture recognition, *IEEE Transactions on Cognitive and Developmental Systems*, **40**: 2021, doi: 10.1109/TCDS.2021.3126637.

25. T. Xianlun, Y. Zhenfu, P. Jiangping, H. Bohui, W. Huiming, J. Li, Selective spatiotemporal features learning for dynamic gesture recognition, *Expert Systems with Applications*, **169**: 4499, 2021, doi: 10.1016/j.eswa.2020.114499.

26. EgoGesture Dataset, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, http://www.nlpr.ia.ac.cn/iva/yfzhang/datasets/ego gesture.html.

Received March 16, 2022; revised version May 3, 2022;
accepted May 16, 2022.