Received: October 4, 2020. Revised: November 5, 2020. 316

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Semantic Deep Learning to Translate Dynamic Sign Language

Eman K. Elsayed[1,2] Doaa R. Fathy[1]*

[1]*Faculty of Science, Al-Azhar University (Girls branch), Cairo, Egypt*
[2]*Masr University for Science & Technology, Collage of Information Technology, Cairo, Egypt*
* Corresponding author's Email: doaaelzalbany@azhar.edu.eg

Abstract: Dynamic Sign Language Recognition aims to recognize hand gestures of any person. Dynamic Sign Language Recognition systems have challenges in recognizing the semantic of hand gestures. These challenges come from the personal differences in hand signs from one person to another. Real-life video gesture frames couldn't be treated as frame-level as a static sign. This research proposes a semantic translation system for dynamic hand gestures using deep learning and ontology. We used the proposed MSLO (Multi Sign Language Ontology) in the semantic translation step. Also, any user can retrain the system to be a personal one. We used Three-dimensional Convolutional Neural Networks followed by Convolutional long short-term memory to improve the recognition accuracy in Dynamic sign language recognition. We applied the proposed system on three dynamic gesture datasets from color videos. The recognition accuracy average was 97.4%. We did all the training and testing processes on the Graphics Processing Unit with the support of Google Colab. Using "Google Colab" in the training process decreases the average run time by about 87.9%. In addition to adding semantic in dynamic sign language translation, the proposed system achieves good results compared to some dynamic sign language recognition systems.

Keywords: Dynamic sign language recognition, Deep learning, Cloud, Semantic, Multi sign Language ontology (MSLO).

1. Introduction

According to the World Health Organization, about 466 million people worldwide have disabled hearing and by 2050, this number will increase to 900 million people [1]. Sign Language has been improved and standardized by many different countries and different cultures resulting in different standards. such as the American Sign Language (ASL) [2], British Sign Language (BSL), African Sign Language, Chinese Sign Language, Portuguese Sign Language, Arabic Sign Language, Egyptian Sign Language, and many more.

Unfortunately, there is no one standard form of sign language and it varies from region to region. Humans know each other by conveying their ideas, thoughts, and experiences to others. So, in addition to using standard sign languages, any deaf or dumb person can have his signs according to his lifestyles, customs, and traditions.

Deaf people have difficulties when communicating with hearing people via writing. Most hearing people also encounter problems with understanding specific gestures, in which Deaf usually writes. Paper and pen are not enough for effective communication.

The sign language recognition (SLR) process involves mainly two tasks, namely, isolated (word-level) and continuous (sentences-level). The isolated tasks are static and dynamic sign language recognition. SLR systems can be used in applicable places like the Industrial-Internet of Things (I-IoT) for Human-Computer Interaction (HCI), Human-Robot Interaction, smart homes for controlling electronic devices using hand gestures. SLR systems can be used in some public places like hospitals, police stations, banks, and other places. The SLR systems can also be used in educational institutions, training centers, and special education centers for especially abled children and many more places. Action and gesture recognition systems can be used in socially assistive robotics, alternative computer

interfaces, immersive game technology, virtual controllers, affective computing, and remote control.

However, the processing power of a person's device may be inefficient for SLR systems' training process and users can't interact with the system from any device. To solve these issues, we have connected the application to a cloud-based framework where we can delegate all heavy work to powerful resources on the cloud and we can accelerate deep learning applications. Google Colaboratory [3] (a.k.a. Colab) is a cloud service based on Jupyter Notebooks to disseminate machine learning education and research. It is free-of-charge access to a robust GPU and provides a runtime fully configured for deep learning. This paper focuses on Dynamic Arabic sign language recognition in user-dependent mode (gestures of one signer). Where there is no standard Arabic sign language, but many variations exist as many as the Arabic-speaking countries. Different Arabic sign languages share only the same signs for alphabets [4]. Other researchers used Ontology in improving sign language translation. Research [5] proposed Multi Sign language Ontology (MSLO). MSLO is a linguistic Ontology that takes the advantages of WordNet and multilingual WordNet on the sign language domain. in MSLO, the WordNet relations are extended to deal with sign language.

The contributions of the current work can be summarized as follows: 1) we suggest using ontology to add the semantic in dynamic gesture recognition. The semantic is added in this proposed system by connection with MSLO and enabled any user to retrain the system using his signs.

2) We used deep learning in the classification process, where Three-dimensional Convolutional Neural Network, then the Convolutional long short-term memory (3DCNN-ConvLSTM) is applied to classify dynamic sign gestures videos' for one user and multiple users. 3DCNN network is used to learn short-term spatiotemporal features of gestures followed by the ConvLSTM network to learn long-term spatiotemporal features. We introduced batch normalization and dropout layers to the ConvLSTM structure. We also tested the performance of our system in predicting unseen gestures to the model.
3) All the training and testing were done on GPU with the support of the Google Colab environment. We run the system in the cloud to solve the problem of inefficient user devices.

This paper is organized as follows, section 2 looks over the literature review in this research area. Section 3 touches on the proposed system and section 4 shows the implementation. Section 5 discusses Experimental results and comparison with others. Finally, section 6 outlines the conclusions.

2. Literature review

Because of the importance of gesture recognition in computer vision and pattern recognition fields, many research works have been done in this context. Researches were done in dynamic ArSLR such as [6], proposed continuous Arabic sign language recognition in user-dependent mode.

A survey on SLR using different classifiers is proposed in [7] to review the best classifier model for sign language recognition (SLR), focused mainly on deep learning techniques and Arabic sign language recognition systems. Different types of classifiers are used for the performance and the deep learning-based classifier was performing better than all the various classifiers in terms of accuracy of sign language recognition [7]. 3DCNNs have been proposed in [8], which use 3D convolutions and 3D pooling to capture discriminative features along both spatial and temporal dimensions.

Long Short-Term Memory (LSTM) inherits most of the Recurrent Neural Network (RNN) models' characteristics and solves the problem of gradient disappearance caused by gradual reduction of gradient backpropagation process, which is often used to predict the sequences' time characteristics [9]. Also, LSTM often neglects the spatial features extraction of feature maps in the sequence's prediction, that is may affect the final spatial and temporal feature information.

The ConvLSTM is a network structure proposed according to convolution operation and LSTM. It extracts spatial features like Convolutional Neural Network (CNN), and model according to time series like LSTM [10]. We organize the literature review into two subsections: some related works on dynamic gesture recognition using deep learning and video classification using deep learning.

2.1 The dynamic gesture recognition using deep learning

With the rapid growth of deep learning and powerful hardware like GPU, deep neural networks (DNNs), and in particular convolutional neural networks (CNNs) have demonstrated more and more superiority in many video classification and prediction tasks. Using CNNs in computer vision is due to their ability to learn rich mid-level image representations as opposed to handcrafted low-level features. So, a lot of researchers have successfully extended the CNNs model for gesture recognition in videos [11].

In this section, we briefly review some of the recent deep learning studies in dynamic SLRs. In many new studies of gesture recognition, people used

3DCNN to extract the temporal and spatial features of video [12], [13]. Authors in [14] have used 3D Convolutional Neural Network (CNN) to detect 25 signs from Arabic Sign Language (ArSL) dictionary. The recognition system is fed with data from depth maps. The input of the system is video, and these videos are divided into frames that will be downsampled and the priority was calculated using the scoring algorithm. The system used the words video stream and gets a normalized depth input. The architecture extracted spatial-temporal features from the input. The experimental results showed the success of the 3D deep architecture. The system accuracy for observed data was 98% and the average accuracy for the new data was 85%.

In [15], 2-ways sign language translators for the Arabic language, which developed a translation of the text to sign and vice versa. CNN along with LSTM is used to perform Arabic dynamic signs to text translation. The ConvLSTM model was evaluated based on the research collected dynamic gestures and the accuracy was 88.67%. Zhang and Li in reference [16] proposed MEMP network. 3D CNN and ConvLSTM are mixed several times to extract and predict the video gesture information multiple times, to get higher accuracy. The MEMP Neural Networks has achieved high accuracy in LSA, IsoGD, and SKIG datasets results are compared with others.

2.2 Video classification using deep learning

Video classification is more challenging compared to static images because of the difficulties of capturing both the spatial and temporal information of consecutive video frames. Authors in reference [17] proposed the 3D convolution operator to compute features from both spatial and temporal. 3D CNN is combined with LSTM to classify video. videos should be divided into several picture sets of frames firstly, and then these frames were operated by 3DCNN in time and space according to the convolution kernel of a certain size. The processed feature set was combined to predict the input of LSTM, and the accuracy can get [18].

Also, according to the high recognition rate of the proposed neural network with an alternative fusion of 3DCNN and ConvLSTM, [16], we build a deep learning model consists of 3DCNN layers followed by ConvLSTM layer.

The Rectified Linear Unit (ReLU) as a simple nonlinear activation function is used to speed network training and reduce the overfitting phenomenon [19]. ReLU can improve the learning speed of neural networks at different depths. This means that using the ReLU activation function can avoid the vanishing gradient problem [20]. Also, the batch normalization (BN) method [21] was utilized to allow the model to use much higher learning rates and less concerned about the initialization to accelerate the training process. 'Dropout' is used to reduce the occurrence of over-fitting, which can reach the effect of regularization to a certain extent [22].

2.2.1 Three-dimensional convolutional neural network (3DCNN)

3DCNN is an extension of CNN by increasing the dimension of the convolution kernel. 3D CNN is good at extracting video features [18]. 3DCNN extracts the entire video's spatial-temporal features to get more comprehensive information [8]. It uses the 3D convolution kernel to extract local spatiotemporal neighborhood features, which is suit for the data format of the video. The 3DCNN formula [18] is represented as follows:

$$v_{ij}^{xyz} = ReLU(b_{ij} + \sum_m \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} \upsilon_{(i-1)m}^{(x+p)(y+q)(z+r)}) \quad (1)$$

Where '*ReLU'* is the activation function of the hidden layer. v_{ij}^{xyz} represents the current value at position *(x, y, z)* in the i-th and j-th feature graph sets. b_{ij} represents the bias of the i-th layer and the j-th feature map. w_{ijm}^{pqr} represents the *(p, q, r)*th value of the kernel connected to the mth feature map in the previous layer. R_i is the size of the 3D kernel along the temporal dimension, where *Pi*, *Qi* represents the height and width of the convolution kernel respectively.

2.2.2 Convolutional LSTM neural network (ConvLSTM)

The ConvLSTM is designed specifically for spatial-temporal sequence prediction problems. Compared with traditional LSTM, the ConvLSTM can better extract spatial and temporal features of feature graph sets [10]. That is because ConvLSTM can consider the spatial information of a single feature map when it processes and predicts the events of time series. So, timing problems in dynamic gesture recognition can be solved using ConvLSTM more effectively. The equations of ConvLSTM [10] are formulated in the flowing equations.

$$i_t = \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i) \quad (2)$$

$$f_t = \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f) \quad (3)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ tanh(W_{hc} * H_{t-1} + W_{xc} * X_t + b_c) \quad (4)$$

$$O_t = \sigma(W_{xO} * X_t + W_{hO} * H_{t-1} + W_{cO} \circ C_t + b_O) \quad (5)$$

$$H_t = O_t \circ tanh(C_t) \quad (6)$$

Where $X_1, X_2, \ldots, X_t$ are the inputs, $C_1, C_2, \ldots, C_t$ are cell outputs, and $H_1, H_2, \ldots, H_t$ are the hidden states. The gates i_t, f_t, and O_t are three-dimensional tensor of ConvLSTM respectively. The last two dimensions (rows and columns) are spatial. The operator '$*$' denotes the convolution operator and '$\circ$' denotes the 'Hadamard product'.

Here, the dropout layer and batch normalization layer are added to the ConvLSTM.

On the other side, Ontology is used in expanding semantic knowledge of the concepts in the image processing domain such as the static image of sign language translation domain [5] and image semantic segmentation [23].

3. Proposed system

We need to rich Deep learning output by semantic, so we proposed Dynamic Gesture Deep Semantic Translation System (DGDSTS) that merges ontology with deep learning. We used the Multi-Sign Language Ontology (MSLO) that was proposed in [5]. MSLO is a semantic layer in deep learning to produce Semantic deep learning (SDL). MSLO is a linguistic Ontology that takes the advantages of WordNet and multilingual WordNet on the sign language domain. MSLO extends the WordNet relations to deal with static sign language and solve some of the sign language challenges. MSLO solves three of these challenges: each sign means a bag of words (synonyms), one meaning has different signs and vice versa, and each person has his-self way to sign any word in any language.

We ameliorate the Sign Language Semantic Translation System (SLSTS) to be a dynamic (DGDSTS) system and more accurate by improving the deep learning model to get more accurate results in case of video sign language recognition.

With the success of technology in the life of deaf and dumb persons, they need to manage their apps and manage gestures of their daily life. Each person may have private signs that simulate his behavior to express feelings and words. Based on the semantic concept we suggest that the user can build his apps

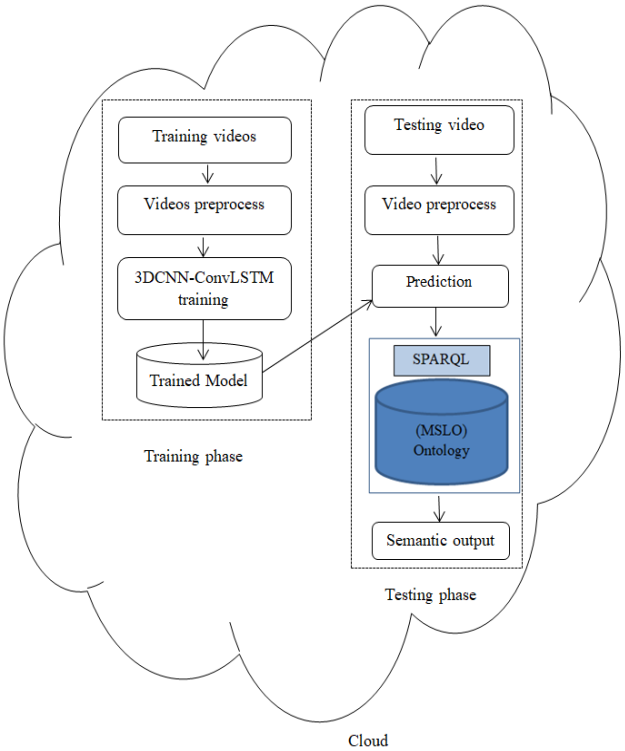

Figure. 1 The proposed DGDSTS system architecture

training gestures. So that the user can manage the dynamic gesture recognition system and record his dependent gestures. First time using the system, the user can create and update the system data. In case of creating a new model, he should record video gestures for training data then preprocess and training to get his trained model. He can input his dynamic gestures for preprocessing and classified using the trained model then semantic translation using SPARQL and MSLO to get the semantic output.

Fig. 1 displays the DGDSTS system architecture where the proposed system has two phases: the training and the testing. Training phase: training the model to extract features of gestures. System training consists of several processes: data preparation, video preprocess then training the 3DCNN+ConvLSTM model. Testing phase: system testing where data preprocess like in the training phase, the saved network parameters are loaded to predict new video gestures. Merging MSLO ontology allows the user to get the semantic output. The details of the processes are as follows:

3.1 Prepare the dataset

In this experiment, we used three color video gesture datasets to evaluate the performance of the proposed system:

Firstly: LSA64 [24] (Argentinian Sign Language) to evaluate our system performance. This dataset consists of 64 classes and each class has 50 RGB

video gesture samples. Therefore, the LSA64 dataset includes 3200 RGB videos and 10 non-expert subjects repeated 64 different LSA signs five times. Each video has a resolution of 1920 × 1080, 60 frames per second. We used 40 vocabularies each word is performed by 10 signers and each signer performed 5 times. So, it would be 2000 videos in total. These 2000 samples were shuffled and non-random 80% of them were picked for training and 20% were used for testing. Videos of each signer split to train and test in the present 8:2. We can't use all 64 classes because there was not enough memory to train all these videos.

Secondly: we select video samples performed by one signer from the LSA dataset includes 320 RGB videos. We divided these samples into training and testing samples, 60% of video samples were used for training, and the rest 40% were used for the testing process. We experiment with one user data such that the user can use the system individually to create his dataset gestures which can simulate his behavior to express particular feelings and words.

Thirdly: we needed to apply the proposed system and test the semantic translation in Arabic dynamic sign language. The dataset with our specification is not available publicly. So, we created a gestures dataset consists of 11 dynamic Arabic gestures of one signer, and each sign repeated 25 times. Those Arabic dynamic gestures were: مال (Money), عمل (Work), قديم (Old), جديد (New), يحفظ (Save), يحذف (Delete), ملف (File), رسالة (Massage), محادثة (Conversation), أصم (deaf), and يترجم (translate). Video samples were recorded using python OpenCV Camera by one signer in many sessions in normal light and normal environment. We divided these video gestures into training and testing, 80% of the data used to train the model, and the rest 20% were used in testing.

3.2 Data pre-process

The dynamic gesture dataset is generally composed of many videos. Video's resolution and time length may be different, so each gesture video in the dataset needs to be preprocessed. Videos will be split into a set of 16 consecutive frames, and then all frames need to be resized to one dimension 32 × 32 for the LSA40 classes' dataset and a one-signer-LSA dataset which improves the computation cost speed of model training, where The Google Colab service provides 12.72 GB of RAM. Our dataset frames resized to one dimension 64 × 64. The frames were arranged in similar folders as the videos, with folder names specifying the gesture class label. Also, in the testing phase when the user uploads the video to be recognized, the video is preprocessed then classify

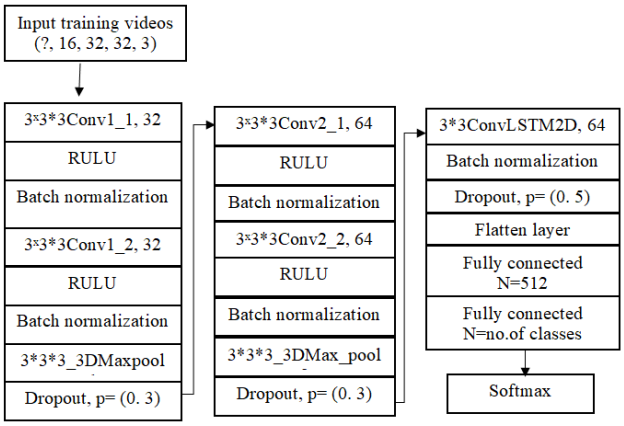

Figure. 2 The overall architecture of the 3DCNN-ConvLSTM model

using the pre-trained model and the result is displayed to the user.

3.3 System training

3.3.1 The 3DCNN-ConvLSTM model architecture

In this section, we will describe the internal structure of the 3DCNN+ ConvLSTM model. For the dynamic gesture recognition process, we proposed a 3D convolution neural network (3DCNN) followed by Convolutional long short-term memory (ConvLSTM) network as a feature extraction model. Fig. 2 illustrates the architecture of the 3DCNN-CONVLSTM model. The input layer is a stack of continuous video sign frames that is resized into 16×32×32×3. The architecture consists of four 3D convolutional layers with different filters 32, 32, 64, and 64 respectively. But the same kernel size 3×3×3. Then one layer of ConvLSTM with 64 sizes of the unit added. Each 3DCNN layer is followed by a ReLU layer and a batch normalization layer. Each two consecutive 3DCNN layers followed with 3D Max Pooling layer and dropout layer. Dropout layers were added with the values of 0.3 and 0.5. The output probability is implemented with a fully connected layer with 512 and followed by Softmax activation function which has many output units corresponding to the number of video signs classes.

3.4 System testing

3.4.1 Prediction of individual sign video

The test video is captured, split into 16 consecutive frames then fed to the trained/learned model for classification. The probability score of each frame is calculated by using the features learned from the model. Then the prediction of 16 frames is given as the input to the majority voting schema which gives the predicted label of the video sequence

based on the probability score of all frames. The formula of the majority voting is given in Eq. (7).

$$Y=mode\{C(X1), C(X2), C(X3), ..., C(X16)\} \quad (7)$$

Where Y is the class label of sign gesture video, $X1, X2, ..., X16$ represents the frames extracted from the tested video. $C(X1), C(X2), C(X3), ..., C(X16)$ represents the class label predicted for each frame.

3.4.2 The connection between the proposed system and MSLO ontology for semantic translation

After loading trained network parameters and recognize video gestures, the output used to search MSLO using SPARQL. SPARQL is used to predict semantically the corresponding meaning. SPARQL is used to access the RDFs inside the ontology by using a SELECT statement as a query. SPARQL query has a standard syntax and depends on using variables that contain the predicate, subject, and objects for RDF. The system would detect the correct meaning according to the user location it will suggest the class of interpretation and define the specific text of meaning.

4. Implementation

The semantic sign language recognition system was implemented on an Intel Corei7 CPU with 8 GB RAM and windows10. The programming language used for implementing the system was Python, using the Anaconda environment, Spyder editor. Deep learning libraries were including Keras along with TensorFlow. Python OpenCV library was used for handling data and preprocessing and rdflib library for working with RDF. The MSLO implemented using Protégé as an Ontology editor.

Since the 3DCNN+convLSTM requires a higher computational complexity for video classification. Personal storage may be limited to train deep learning models. Therefore, we also applied the training and testing processes on Google Colaboratory [3] as a cloud service with the same libraries that we used in anaconda. The datasets were mounted to the Google Colab using Google Drive. Google Colab is a cloud service based on Jupyter Notebooks to disseminate machine learning education and research. It is free-of-charge access to a robust GPU and provides a runtime fully configured for deep learning. Training process on GPU with support of the Cloud decreases the training time. We suggested using the system in Cloud to enable the user to arrive at his system remotely and save users' device storage.

5. Experimental results and discussion

The proposed DGDSTS system is considered a prototype of one user dynamic sign language recognition system. The recognition accuracy was held to find out the performance of the trained models. In our experiments, we will show the recognition accuracy of the three used datasets according to the proposed 3DCNN+ConvLSTM model at different iteration values and according to 3DCNN model at the same iteration that the system gets high recognition accuracy performance.

Deep learning models have many parameters that affect the model progress and performance. We will discuss the effect of iteration numbers on our network performance. Iteration number is one of the most important hyperparameters in modern deep learning systems. Practically, using a small iteration number in training the model allows more computational speedups from the parallelism of GPUs. On the other hand, using larger iteration numbers led to higher testing accuracy than small iteration numbers but the training process slower than the small iteration number. Batch size and epochs number can mainly vary with the size of the model training dataset.

Table 1 summarizes the accuracy achieved by the 3DCNN+ConvLSTM architecture at different iterations. Despite using a small batch size, there is a clear change accuracy with the number of iterations; the biggest epoch still achieves the best accuracy. In the case of model training for 100 epochs, it achieves the highest recognition accuracy in the three datasets. The model achieves 98.5%, 99.2% and 94.5% recognition accuracy respectively. Validation accuracy was 100% and approximately 100% training accuracy in all cases.

Despite the same environment of the two datasets: LSA 40 classes and one-user-LSA 64 classes, the recognition accuracy of a dataset of on

Table 1. Results of the used datasets at different epochs

Epoch	3DCNN+ConvLSTM		
	Our dataset	One-signer LSA	LSA40
10	81.8	92.9	98
20	85.4	97.6	98
30	90.9	97.6	98
40	87.3	95.3	98.2
50	92.7	96	98.2
60	85.4	96.8	98.2
70	87.2	96.8	98.2
80	89	96.8	98.2
90	93.7	99.2	98.5
100	94.5	99.2	98.5

signer is higher than the recognition accuracy of a dataset of multiple signers.

5.1 Experiments on LSA64 dataset

Training samples were used to tune the model over batch size 32 samples and with different numbers of iterations. 25 percent of the training samples were used for model validation. The performance on the training and validation datasets as illustrated in Fig. 3, it is clear that the system is performed well on the training dataset. The samples were not diverse enough for system training; the training accuracy was 100%. 20% of the LSA64 dataset used in testing the trained model, the best recognition accuracy rate was 98.5% on iteration value 100 epochs.

5.2 Experiments on the one-signer-LSA dataset

In this case, the training and testing samples were selected from the LSA dataset performed by one signer. the training samples were used to tune the model over batch size 16 samples for the one-signer-LSA dataset and with different numbers of iterations.

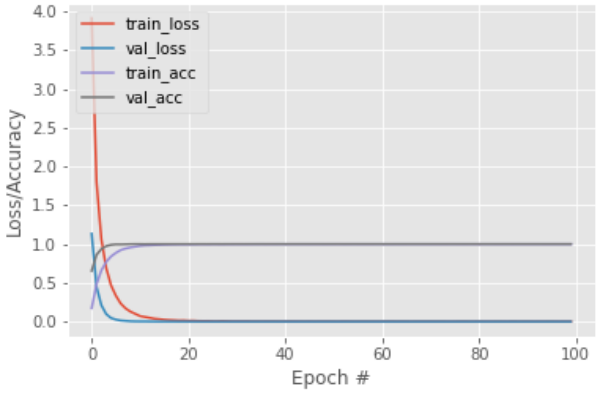

Figure. 3 Training and validation accuracy of 40 class-LSA dataset's model

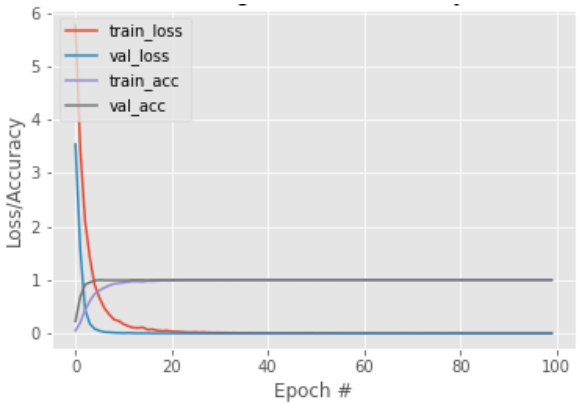

Figure. 4 Training and validation accuracy of one-signer-LSA dataset's model

25 percent of the training samples were used for model validation. The performance on the validation dataset as illustrated in Fig. 4, it is clear that the system was overfitted on the training dataset. The samples were not diverse enough for system training. However, the training accuracy was 100%. 40% of the one-signer-LSA dataset used in testing the trained model, the recognition accuracy rate was 99.2%. It is showed that the recognition accuracy in the signer dependent mode higher than in signer independent mode.

5.3 Experiments on our dataset

Training samples were used to tune the model over batch size 8 samples for the one-signer dataset and with different numbers of iterations. 25 percent of the training samples were used for model validation. In this case, system training accuracy was 100%, recognition accuracy was 94.5%.

In terms of the training time of the proposed model, the training time on Google Colab is higher run-time than the local machine. Table 2 shows the best experimental results for the three datasets and the execution time in case of training on windows using the Anaconda environment. The improvement shown indicates the performance comparison between 3DCNN+ConvLSTM model compared to the 3DCNN model.

We noted that the proposed 3DCNN+ConvLSTM architecture achieved better recognition accuracy in all scenarios as compared to the results of 3DCNN architecture that achieves recognition accuracy 97.7%, 96.8%, and 85.4% respectively.

Table 2 shows that data preprocess and training took 9 min on Google Colab and 2 hours on windows using the Anaconda environment to complete training of the LSA one user sign language dataset i.e. the run time decreases approximately by 92.5%. Also, preprocess and training of LSA forty vocabulary dataset took about 2h on Google Colab and about 14 hours on windows i.e. the run time decreases approximately by 85.7%. Our dataset's data preprocess and training took 9 min on Google Colab and 2 hours on windows using an anaconda environment also i.e. the run time decrease approximately by 85.7%. We can say that the average time decreases approximately by 87.9%.

5.4 Comparison with other works

To construct a comparison between the proposed DGDSTS system and some works those are displayed in related works. Table 3 shows this comparison. The comparison is based on some criteria that represent the main features in the research field. These features

Table 2. Results of the 3 datasets at 100 epoch

Dataset	No. signer	No. classes	No. testing videos	No. training videos	Classification	3DCNN + ConvLSTM Recogni	Execution time on windows	Execution time on cloud	3DCNN Recognition accuracy
LSA [24]	10	40	400	1600	%100	98.5%	14 h	2 h	97.7%
LSA one signer	1	64	128	192	%100	99.2%	2 h	9 min	96.8%
Our recorded gestures	1	11	55	220	%100	94.5%	2 h	9 min	85.4%

Table 3. Comparison results

Ref. #, year	Semantic translation	Personal availability	Dataset	Computing environment	Recognition accuracy average	Method
The proposed DGDSTS	Yes	Yes	(dynamic) LSA40, LAS one-signer, collected dynamic Arabic gestures	Cloud	97.4%	3DCNN+ConvLSTM
E. K. Elsayed and D. R. Fathy. [5] 2020	Yes	Yes	(Static) pre-made ArSL dataset and collected static gestures dataset	Personal computer system (anaconda)	91.59%	CNN
X. Zhang and L. Xiaoqiang. [16] 2019	No	No	(dynamic) LSA64, SKIG and Chalearn 2016	Personal computer system	90.2%	an alternate fusion of a 3D CNN and ConvLSTM
ElBadawy et al. [14] 2017	No	No	(dynamic) 25 video signs from ArSL dictionary	Personal computer system	85%	3DCNN

are the semantic translation, the personal availability, the dataset type (static or dynamic), type of computing environment, recognition accuracy average, and the deep learning method. Compared to the existing works and experiment results, we find that our system performs better than the others in semantic translation and classification of dynamic sign language. We take the advantages of the personal availability, semantic translation of dynamic sign language, high recognition accuracy, and availability of using the system in a Cloud Computing Environment using Google Colab. As shown in table 3 other systems used personal computer systems. Cloud helps us to save storage, decrease the time of the training process, and give users the ability to access systems remotely.

We used LSA 40 classes' dataset and the accuracy was 98.5%. But reference [16] used LSA 64 classes' dataset, and the accuracy was 99.063% when it used an alternative fusion of a 3D CNN and ConvLSTM. Reference [5] used the traditional CNN model in addition to semantic in Arabic static sign language translation but get 91.59% recognition accuracy. As we show in our experiments, the proposed 3DCNN+ConvLSTM architecture achieved better recognition accuracy in all scenarios as compared to 3DCNN architecture. Reference [14] has an accuracy 85% where they used 3DCNN architecture. So we can say that the height of our average accuracy because of the 3DCNN+ convLSTM performed well in case of user-dependent and non-dependent user dynamic sign language recognition.

6. Conclusion

This paper presented the first time to develop dynamic gesture recognition using deep learning and ontology with Cloud as a Computing environment. In this paper, we proposed a DGDSTS system for dynamic gesture semantic translation. The system combined deep learning with Ontology thus taking the advantages of both. Deep learning is a powerful artificial intelligence tool in video classification. The combination of 3DCNN and ConvLSTM model extracts the spatial-temporal features of the gesture sequence, especially with the dynamic gesture recognition. We used the advantages of semantic

through merging output with MSLO that solve some of the sign language challenges.

We evaluated the proposed system by applying it in three datasets, one of them is the user-dependent dynamic Arabic signs dataset that we collect in a normal environment. The three datasets exhibited excellent performance; the system training accuracy was 100%. The recognition accuracy was 98.5%, 99.2%, and 94.5% respectively. 3DCNN+ConvLSTM achieved a good result with both signer-dependent and signer-independent datasets compared with 3DCNN.

We practice the advantages of cloud computing by using Google Colab, where the proposed DGDSTS system average training time decreased approximately by 87.9%. The system saved the storage of the user device and can be managed remotely.

In the future, this system may be used in continuous semantic recognition of sign language in a mobile chat system. We plan to include the development of the Arabic sign language semantic using MSLO, Improvements in the deep learning framework will also be made, which will include a higher degree of user modalities such as facial expression, head, and body actions.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Eman K. Elsayed gives the idea of the system, Doaa R. Fathy; software and designed the experiments, Eman K. Elsayed, and Doaa R. Fathy; formal analysis, investigation, resources, data preparation, Doaa R. Fathy; writing—original draft preparation, Eman K. Elsayed; writing—review and editing, Eman K. Elsayed and Doaa R. Fathy; supervised the study, analyzed the results, and verified the findings of the study.

References

[1] World health. Deafness and hearing loss organization, https://www.who.int/healthtopics/hearing-loss, [Online] [accessed: 5- 31- 2020].

[2] C. Zhang, Y. Tian, and M. Huenerfauth, "Multi-modality American sign language recognition", In: *Proc. of IEEE International Conf. on Information Processing (ICIP)*, Phoenix, Arizona, USA, pp. 2881–2885, 2016.

[3] Google, Colaboratory: Frequently Asked Questions, https://research.google.com/colaboratory/faq.html, [Online] [accessed: 8- 14- 2020].

[4] S. M. Halawani, "Arabic sign language translation system on mobile devices", *IJCSNS International Journal of Computer Science and Network Security*, Vol. 8, No. 1, pp. 251-256, 2008.

[5] E. K. Eman and D. R. Fathy, "Sign Language Semantic Translation System using Ontology and Deep Learning", *(IJACSA) International Journal of Advanced Computer Science and Applications*, Vol. 11, No. 1, 2020.

[6] N. Tubaiz, T. Shanableh, and K. Assaleh, "Glove-based continuous Arabic sign language recognition in user-dependent mode", *IEEE Transactions on Human-Machine Systems*, Vol. 45, No. 4, pp. 526–533, 2015.

[7] M. Mustafa, "A study on Arabic sign language recognition for differently abled using advanced machine learning classifiers", *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-15, 2020.

[8] D. Tran, L. Bourdev, R. Fergus, L. Torresan, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks", In: *Proc. of the IEEE International Conf. in computer Vision (ICCV)*, Santiago, Chile, pp. 4489-4497, 2015.

[9] D. E. D'Informatique, N. Ese, P. Esent, E. Au, F. Gers, P. R. Hersch, P. Esident, and P. P. Frasconi, "Long short-term memory in recurrent neural networks", *Epfl 2001*, Vol. 9, No. 1, pp. 1735–1780, 2001.

[10] S. Xingjian, Z. Chen, H. Wang, and D. Yeung, "Convolutional lstm network: A machine learning approach for precipitation nowcasting", *Advances in Neural Information Processing Systems*, pp. 802–810, 2015.

[11] M. A. Aghbolaghi, A. Clapes, M. Bellantonio, H. J. Escalante, V. Ponce-López, X. Baró, I. Guyon, S. Kasaei, and S. Escalera, "A survey on deep learning based approaches for action and gesture recognition in image sequences", In: *Proc. of 12th IEEE International Conf. on Automatic Face & Gesture Recognition*, Washington, DC, USA, pp. 476-483, 2017.

[12] F. Sachara, T. Kopinski, A. Gepperth, and U. Handmann, "A. Free-hand gesture recognition with 3D-CNNs for in-car infotainment control in real-time", In: *Proc. of the IEEE 20th International Conf. on Intelligent Transportation Systems (ITSC)*, Yokohama, JAPAN, pp. 959–964, 2017.

[13] C. Lin, J. Wan, Y. Liang, and S. Z. Li, "Large-scale isolated gesture recognition using a refined fused model based on masked res-c3d network

and skeleton lstm", In: *Proc. of 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, Xi'an, China, pp. 52-58, 2018.

[14] M. ElBadawy, A. S. Elons, A. H. Shedeed, and M.F. Tolba, "Arabic sign language recognition with 3d convolutional neural networks", In: *Proc. of IEEE Eighth International Conf. on Intelligent Computing and Information Systems (ICICIS)*, Cairo, Egypt, pp. 66-71, 2017.

[15] T. Agrawal and S. Urolagin, "2-way Arabic Sign Language Translator using CNNLSTM Architecture and NLP", In*: Proc. of the 2020 2nd International Conf. on Big Data Engineering and Technology*, Singapore China, pp. 96-101, 2020.

[16] X. Zhang and X. Li, "Dynamic gesture recognition based on MEMP network", *Future Internet*, Vol. 11, No. 4, p. 91, 2019.

[17] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 1, pp. 221-231, 2012.

[18] N. Lu, Y. Wu, L. Feng, and J. Song, "Deep learning for fall detection: Three-dimensional CNN combined with LSTM on video kinematic data", *IEEE Journal of Biomedical and Health Informatics*, Vol. 23, No. 1, pp. 314-323, 2018.

[19] G. E. Dahl, T. N. Sainath, and G. E. Hinton. "Improving deep neural networks for LVCSR using rectified linear units and dropout", In: *Proc. of IEEE Int. Conf. Acoust., Speech Signal Process.*, Canada, pp. 8609–8613, 2013.

[20] H. Ide, and T. Kurita, "Improvement of learning for CNN with ReLU activation by sparse regularization", In: *Proc. of the 2017 International Joint Conference on Neural Networks (IJCNN*), Anchorage, USA, pp. 2684–2691,14–19 May 2017.

[21] S. Loffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", In: *Proc. of the 32nd International Conf. on Machine Learning*, arXiv 2015, arXiv: 1502.03167. Available online: https://arxiv.org/abs/1502.03167 (accessed on 5 -8- 2020).

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "A. Dropout: A simple way to prevent neural networks from overfitting", *The Journal of Machine Learning Research*, Vol. 15, No. 1, pp. 1929–1958, 2014.

[23] E. Elsayed and M. Aly, "Hybrid between Ontology and Quantum Particle Swarm Optimization for Segmenting Noisy Plant Disease Image", *International Journa of Intelligent Engineering and Systems*, Vol. 12, No. 5, pp. 299-311, 2019.

[24] F. Quiroga, F. Ronchetti, C, Estrebou, L. Lanzarini, and A. Rosete, "Sign language recognition without frame-sequencing constraints: A proof of concept on the argentinian sign language", In: *Proc. of Ibero-American Conf. on Artificial Intelligence. Springer*, Cham, pp. 338–349, 2016.