

Dynamic hand gesture recognition of Arabic sign language by using deep convolutional neural networks

Mohammad H. Ismail, Shefa A. Dawwd, Fakhruddin H. Ali

Department of Computer Engineering, College of Engineering, University of Mosul, Iraq

Article Info

Article history:

Received Aug 28, 2021

Revised Nov 20, 2021

Accepted Dec 9, 2021

Keywords:

Arabic sign language
Convolutional neural network
Deep learning
dynamic hand gesture
Multi-model

ABSTRACT

In computer vision, one of the most difficult problems is human gestures in videos recognition Because of certain irrelevant environmental variables. This issue has been solved by using single deep networks to learn spatio-temporal characteristics from video data, and this approach is still insufficient to handle both problems at the same time. As a result, the researchers fused various models to allow for the effective collection of important shape information as well as precise spatiotemporal variation of gestures. In this study, we collected the dynamic dataset for twenty meaningful words of Arabic sign language (ArSL) using a Microsoft Kinect v2 camera. The recorded data included 7350 red, green, and blue (RGB) videos and 7350 depth videos. We proposed four deep neural networks models using 2D and 3D convolutional neural network (CNN) to cover all feature extraction methods and then passing these features to the recurrent neural network (RNN) for sequence classification. Long short-term memory (LSTM) and gated recurrent unit (GRU) are two types of using RNN. Also, the research included evaluation fusion techniques for several types of multiple models. The experiment results show the best multi-model for the dynamic dataset of the ArSL recognition achieved 100% accuracy.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mohammad H. Ismail
Department of Computer Engineering, College of Engineering
University of Mosul, Mosul, Iraq
Email: mohammad.haqqi@gmail.com

1. INTRODUCTION

People often utilize hand gestures to communicate their thoughts and feelings. People who are deaf or hard of hearing have traditionally depended on sign language to communicate. Most ordinary people do not know this language and have difficulties in communication with deaf people. As a result, in order to ease communication and close the gap, an automatic sign language recognition system must be developed. Sign language's organized form of hand movements aids nonverbal communication between deaf and hard of hearing people. Many vocabularies or words in sign languages have a complicated structure, comparable to that of spoken languages. Handshape, position, direction, movement, and facial emotions all contribute to the expression of sign language movements [1]. The difficulties of sign language recognition may be split into two categories: static gesture recognition, which focuses on finger spelling, and dynamic gesture recognition, which includes single words and continuous sentence recognition. For full sentence recognition, most continuous sign language systems utilize an enhanced version of the single word framework [2]. Hand segmentation/detection, handshape feature representation, and sequence classification are three difficulties faced by independent dynamic sign language recognition systems.

The majority of sign language recognition systems have been divided into two categories glove-based and image-based methods [3], [4]. Signers must wear an electronic sensor glove that monitors and

detects hand and finger movements in order to use the glove-based method. The main disadvantage of this method is that the signer must execute the signs while wearing a heavy glove with a pair of sensors. Without the need of gloves or sensors, the vision-based method records the hand gesture in the form of sequential or fixed images by the camera. Although there are numerous difficulties such as background variations, skin colour, lighting conditions, and the characteristics and settings of the camera, this method is more suitable for the deaf and mute in their everyday lives and simpler for the signer [4]. The segmentation of the hands and fingers, like this technique, requires a lot of processing. To make the segmentation process easier, the signer may be asked to wear coloured cotton gloves.

Recently, a method based on the Microsoft Kinect (MK) device was presented. MK is a motion sensing device that Microsoft initially designed for better user experience during video gaming and entertainment [5]. The Kinect V2 sensor consists of colour and infrared (IR) cameras. The colour camera outputs a high-resolution red, green, and blue (RGB) video stream with a resolution of 1920x1080 pixels. The IR camera captures the modulated infrared light sent out by the IR projector/emitter to output depth images/maps that determine the distance between the sensor and each point of the scene based on the Time-of-Flight (ToF) and intensity modulation technique. The depth images are encoded using 16 bits and have a resolution of 512x424 pixels [6]. In addition, the Kinect V2 sensor provides information about the signer's skeleton through 25 joint points, and it is equipped with a microphone array to capture sounds. This device's data has been examined for different study fields and it has been utilized in limited scope for gesture recognition of Arabic sign language (ArSL). Therefore, we utilized the MK device to recognize ArSL in this study.

The primary purpose of this research is to prepare the data of ArSL, amounting for each of the RGB and the depth to 7,350 videos of sign language words. Then, proposing single models and choosing the best one to recognize ArSL based on this dataset. Then evaluation is done employing different pre-trained models in the architecture of the chosen model to recognize ArSL on our dataset as a single model and then as a multi-model. As well as evaluation of the fusion model's method is done.

2. RELATED WORKS

Sign language (SL) is a natural and efficient way for hard of hearing or deaf individuals to communicate with others in their society. Interacting with them necessitates learning sign language or another type of communication. Sign language has become of great interest to researchers with the rapid development of multimedia communication technologies to improve social communication among its users. One of the main trends in sign language recognition mentioned is vision-based systems which using depth cameras to capture colour or depth images/videos in a more natural environment. Several machine learning algorithms have been created, to analyze and categorize video data.

The minor details of ArSL were dealt with using a features extractor with deep behavior for ArSL Recognition [7]. Utilized 3D-convolutional neural network (CNN) to identify 25 gestures from an Arabic sign language dictionary. Data from depth maps was input into the recognition system. For observed data, the system was 98% accurate, while for new data, it was 85% accurate on average. Masood *et al.* [8] used approximately 2300 videos for 46 gesture categories to propose a vision-based system to translate isolated hand gestures from the Argentinean sign language, with an accuracy of 95.217%. They utilized two methods to categorize the data: CNN for spatial features and recurrent neural network (RNN) for temporal features. The real-time system is described in [9] for an automated ArSL recognition system based on the Kinect sensor that compares signs and recognizes 30 isolated words from standard ArSL signs using the dynamic time warping algorithm. The system obtains a signer-dependent recognition rate of 97.58% and a signer-independent recognition rate of 95.25%.

A multi-model dynamic sign language recognition technique based on a deep 3-dimensional residual ConvNet and Bi-directional long short-term memory (LSTM) networks (B3D ResNet) was reported by Liao *et al.* [10]. Their technique is divided into three parts. The hand item was first located in the video frames. The B3D ResNet was then used to extract spatiotemporal features from the video sequences, and the dynamic sign language was correctly recognized by classifying the video sequences. On the DEVISIGN_D dataset, the experiment got a recognition accuracy of 89.8%, while on the simple linear regression (SLR) Dataset, it got 86.9%. A new dynamic gesture recognition technique was presented by Zhang *et al.* [11]. To extract frame-level motion features for gesture categorization, a motion representation 2D-CNN model was first proposed. The sequential motion information in this model may be represented into a single image. Second, the suggested 3D DenseNet model was built to directly capture spatiotemporal motion information from RGB videos. Finally, the 2D-CNN model's prediction results and the 3D DenseNet model's prediction results were merged to improve performance. Their suggested approach obtained classification accuracy rates of 89.1% and 89.5%, respectively, on the difficult vision for intelligent vehicles and applications (VIVA)-dataset and the UTD-multimodal human action dataset (MHAD) dataset. The Jester 20BN-JESTER dataset

was used in reference [12] to train the 3D-CNN architecture for hand gesture detection by using the 3D-CNN model, which comprises four convolution layers and two fully connected layers one dropout layer, there are 27 categories of hand gestures. They obtained a 90% accuracy model with three days of training.

Hand semantic segmentation, hand shape feature representation, and deep recurrent neural network are all part of a novel framework that uses several deep learning architectures, was presented in [13] for independent ArSL recognition including 23 isolated words. A single layer convolutional self-organizing map was used to extract hand shape features. A deep BiLSTM recurrent neural network recognizes the sequence of extracted feature vectors. The suggested system obtains an average accuracy of 89.5% when tested on the Arabic benchmark dataset. Tran *et al.* [14] presented a novel technique to identify fingers and recognize hand gestures in real-time using an RGB-D camera and a three-dimensional convolution neural network (3D-CNN). Their network included seven convolutional layers, three max pooling layers, and two fully connected layers, and their system had seven hand gestures. They evaluated their 3D-CNN training time of 1 hour and 35 minutes with 92.6% accuracy, and then increased the accuracy to 97.12% using the ensemble model with 15 different 3D-CNNs.

Sarma *et al.* [15] developed a two-stream network to detect and identify isolated dynamic hand gestures with changing form, size, and colour of the hand. The suggested method utilized 3D-CNN to extract motion patterns for gesture classification and 2D-CNN to capture Spatio-temporal information directly from RGB gesture videos. They obtained an accuracy of 92.60% when using only 2D-CNN, 97.30% when using only 3D-CNN, and 99.20% when using only the fusion model. Rastgoo *et al.* [16] presented a deep-based model for effective hand sign recognition by training: first, the single shot detector (SSD) model for hand identification using annotated videos of five online sign dictionaries, and second, a combinational model with a CNN and different spatial features. They performed a comprehensive study of sequence learning utilizing various pre-train models, spatial features, and temporal-based models. They extracted features from still RGB frames using the ResNet50 model, which had an accuracy of 86.32% and a prediction time of 2.58 seconds for each sign.

Elsayed and Fathy [17] presented a system that utilized 3D-CNN and LSTM to enhance dynamic sign language recognition accuracy on three dynamic gesture datasets extracted from colour videos, with an average recognition accuracy of 97.4%. Elhagry and Gla [18] introduced a vision-based system for translating some ArSL gestures into its alternate isolate words. They obtained an accuracy of 90% by using the InceptionV3 CNN architecture and an accuracy of 72% by using InceptionV3 CNN-LSTM. They demonstrated that CNN gives high accuracies for isolated sign language recognition, while CNN-LSTM is an excellent choice for continuous word recognition.

There are various efforts in the present research to construct both single and multiple models to improve the accuracy and performance of the recognition for ArSL. The following were the highlights of this study:

- A large-scale ArSL dataset prepared using Kinect V2 that includes RGB and depth Because of the unavailability and inability to access such as this data.
- There is no evaluation of the different methods of fusion models to recognize ArSL using multi-model fusion in previous researches.
- The model used in evaluating the fusion methods was selected from several proposed single models and from several pre-trained multi-models unified in the method of fusion after evaluating their performance in recognition of the ArSL.

3. EXPERIMENTAL METHODOLOGY

3.1. Dataset

The dynamic dataset was collected using a Kinect camera, and each recorded data contains RGB and Depth video. The dataset consists of 7,350 isolated video samples of sign language words, which show ArSL for a total of twenty meaningful words and one no sign doing other things. The doing other things category is a collection of various activities such as a simple body or head movement. Three hundred fifty recorded videos covered each word gesture. The video data was used for Chinese Sign Language with the same meaning or employed in other meanings in ArSL, which represented 250 videos recorded for each word repeated five times by 50 people. In addition to recording 100 videos for each word repeated 20 times by five people. The recorded video used Python OpenCV camera with variations in surroundings, clothing and lighting. The duration of gesture length was from 1.5 sec to 3 sec. The present study words are: [Nothing, Cheek, Friend, Plate, Marriage, Moon, Break, Broom, You, Mirror, Table, Truth, Watch, Arch, Successful, Short, Smoking, I, Push, stingy, and Long], Figure 1 shows the dynamic words signs samples for ArSL. The data set was divided into three subsets: 85.7% for training, i.e. 6,300 videos of sign language words, 7.14% for validation 525 videos of sign language words and 7.14% for testing, i.e. 525 videos of sign language

words. The datasets were collected using Microsoft Kinect 2.0; the recorded data included 7,350 RGB videos and 7,350 depth videos.



Figure 1. Dynamic words signs samples from unified ArSL dictionary

3.2. Data preprocessing

The dynamic gesture dataset is generally composed of many videos. Video's resolution and time length may differ, so each gesture video in the dataset must be preprocessed. Videos will be split into consecutive frames, and the upper signer body with his hands is cropping from the frame using the capture skeleton. Then the frame is normalized where different users may perform gesture moves at different speeds, and the neural network input should be the same. Since the time length of the recorded video is variable to express a specific word of the sign language, when the videos are converted to frames, it may exceed 80 frames. In some cases, the video's frames include additional frames before and after the sign language, which expresses a specific word. These additional frames that did not express a word were removed. Therefore, 32 frames were used to express a word expressed in sign language. After that, all frames need to be resized to 200×200 . The frames were organized in folders that were identical to those used for the videos, with the folder names indicating the sign class label.

3.3. Data augmentation

Deep learning is connected with millions of images for very strong deep neural networks. The small training image set problem is that although the neural network remembers our training data and can accurately predict the training set's performance, the validation accuracy is low. To avoid overfitting and increase model generalization capabilities, data augmentation was used to solve the dataset issue [19]. To avoid model overfitting and improve learning capabilities, numerous data augmentation strategies are utilized for dynamic sign language: Image normalization, zoom range (1.0, 1.2), width shift range (10%), height shift range (10%), rotation range ($\pm 10^\circ$), and brightness range (0.4-1.2). Also, data was augmented by blurring images using averaging, median, and gaussian for the dynamic sign. And morphological operations erosion and dilation, adding noise salt and paper, and sharpening the dataset's images. Through these operations, the training set is enlarged by around 48 times. All of these operations are applied at random inside the mini-batch input into the model. In this work during training, we performed an online spatiotemporal data augmentation, for spatial augmentations and we apply various data augmentation techniques. And for temporal augmentation, we selected consecutive frames in a range of defined input sizes.

3.4. Proposed models

CNN was used to train the model on spatial features, while RNN was used to train the model on temporal features. CNN succeeds in identifying patterns and applying them to image classification. CNN assumes that the network's input will be an image. Recurrent neural networks (RNNs) utilize the information of sequence itself to perform recognition tasks. Because RNNs contain loops, their output is dependent on the mix of current input and previous output. Because a video sequence includes both temporal and spatial

information, it is hard to classify video. From the frames of the video, the spatial features are extracted, whereas from relating the frames of video with the time the temporal features are extracted. Dynamic gestures involve a series of frames, not just one frame, and these frames will depend on each other over time. Several models are built to recognize gestures, including 3D-CNN or 2D-CNN with RNN. CNN is used to extract the features for each image, and then the sequence of these features is passed to the RNN. A fully connected layer follows the output of the RNN with softmax activation for a classification problem.

For the empirical evaluations, we constructed four deep neural networks. Figures 2 illustrate the models that were constructed and applied to the video dataset. And which they have been trained and tested on this dataset. These models cover all methods of feature extraction and the process for passing those features onto RNN. Model 1: Recurrent convolutional neural network (RCNN), by its name, is a combination of CNN and RNN. RCNN uses a 2D convolutional network to extract spatial features on frames, which was done by using pre-trained architecture such as ResNet50 [20], DenseNet121 [21], VGG16 [22], and MobileNet [23]. Then these features were injected to RNN, which include recurrence relations, meaning that each subsequent output is dependent on a mixture of previous outputs. For RNN, the two methods included in RNN are LSTM and gated recurrent unit (GRU), which are stacked on top of the convolutional network to model temporal dependencies. One, two or three RNN layers were used. Model 2 is an architecture that includes conv2d to extract features and use LSTM to handle sequences between frames. Model 3: 3D-CNN with LSTM the model consists of a 3D-CNN and an LSTM for sequence classification. The 3D-CNN takes blocks of 32*200*200 frames of three channels as input and produces a feature vector. The LSTM takes that vector sequentially for an entire gesture and gives the final prediction. Model 4: Each sign word gesture is represented using 32 consecutive frames, with dimensions 200x200 and three channels. The model consists of six convolution 3D layers, which was used to extract features and the relationship between sequences of frames. Convolution layers are also followed by batch normalization, and a pooling layer follows every two convolution layers for down-sampling by a factor of two. Convolution is performed with filter size 2, padding same, and each Convolution layer followed by ReLU activation. And then, it is followed by three fully connected layers (FCs) for classification.

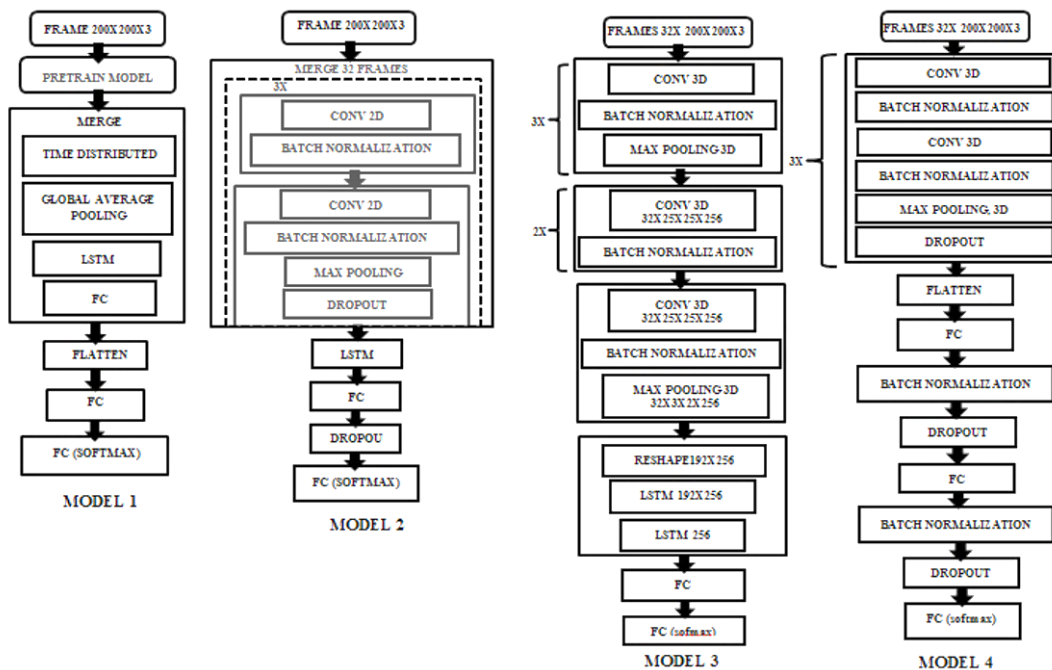


Figure 2. The architecture of proposed models

3.5. Multi-model fusion

The reason for using multi-model data is that complementary information, which includes RGB and Depth, provides a richer representation that may be utilized to produce much better results than single model data. The multi-model fusion research community has made significant progress [24]. Data level fusion, Feature level fusion, and Decision level fusion were the three kinds of multi-model fusion techniques we tested as shown in Figure 3. Data level fusion: The Data level fusion of RGB and depth data include creating

a new matrix for input data with four channels (RGBD). In the presented research, a CNN model was used to process the RGBD input. We used a conv2d for feature extractor to process the RGBD input. And temporal relations of the frames are captured by an LSTM layer, for this fusion used, the model needs four channels as input data. When we used the pre-train model ResNet50, which accepts only three channels input, we added one extra layer conv2D to pass the 4-channel input through this layer to make the output of this layer suitable for ResNet architecture, then LSTM handle sequence between frames, To predict the probability of each of the result classes, three fully connected (FC) layers were utilized, followed by an output layer with softmax/sigmoid activation. Feature level fusion: it concatenates the two separate models in a specific convolutional layer or fully connected layer. Features extraction from RGB and depth streams of a sign frame in parallel is used. It concatenates the output feature maps from two models' final convolutional layers as one input to the following layers. Decision level fusion: it combined the predictions probability results from two separate models [25]. One model takes an RGB video as input, and the other takes the depth video as input. The process is done by collecting the probability of twenty-one class recognition for each of the two models and taking the average for it, then adopting the highest probability to represent the correct distinction for that class.

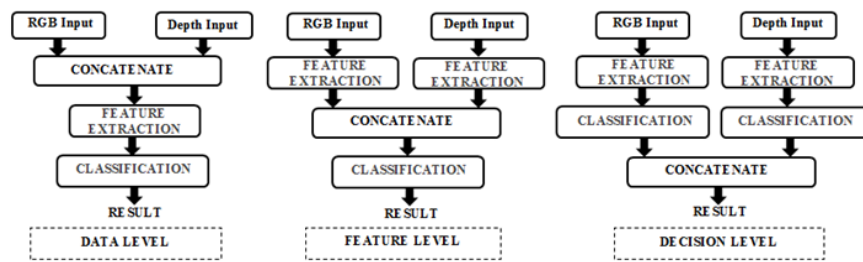


Figure 3. Types of fusion levels

4. RESULTS AND DISCUSSION

For experiment hardware used, we used Kaggle, a cloud service based on Jupyter Notebooks. All the training was done on graphics processing unit (GPU) with the support of the kaggle environment, the kaggle service provides GPU Nvidia P100 which has 16GB VRAM with 3,584 compute unified device architecture (CUDA) cores. Also, we used the Legion Y-540 personal laptop for inference. Its specifications are: Graphics card in this system is Nvidia GeForce RTX 2060, which has 6GB GDDR5X VRAM with 1,920 CUDA cores, processor used is Intel Core i7. Our new large-scale ArSL dataset evaluated our model's performance by randomly choosing the training, validation, and test sets. The evaluation measure is presented in the results tables of our experiments.

4.1. Evaluation metric and hyperparameters

We get defined with some of the terms used in the evaluation scale and hyperparameters in analyzing the results of the present study. Batch size: is the number of samples that are passed to the network at once during training. The GPU memory consumed by the deep learning (DL) model is often unknown before starting training or inferencing, so incorrect choice of neural network architecture or hyperparameters can cause the task to fail to run out of memory. Accordingly, the batch size is used in proportion to the size of the limited GPU memory for consumption; therefore, we usually use the largest batch size possible to our resources [26]. Channels: one and three represent gray and RGB frames, respectively. In object detection, the most frequent unit of time is the frame per second (FPS), the maximum number of frames that the network can handle in a second is indicated by this value. The time it takes to train the network for 20 epochs is called training time. Incorrect recognize testing, validation, and training: For testing, validation, and training, the number of misclassified dataset videos. Total incorrect recognize: the number of videos that have been misclassified. The number of correct classifications divided by the total number of classifications is known as accuracy. For model's performances evaluation, we use the number of misclassified videos, incorrect, as defined,

$$Incorrect = \sum_{i=1}^N f[y(i),p(i)] \tag{1}$$

where N is the total number of samples; y represents the actual label; p represents the predicted label; if $y(i) \neq p(i)$; $f[y(i),p(i)]=0$, otherwise $f[y(i),p(i)]=1$

Inference: Inference is the stage of using a trained model to infer or predict test samples. We measure the inference of each isolated video sign word of each deep neural network (DNN) model of GPU

and CPU. We measure the inference in FPS. It was statistically verified by finding the average number of FPS according to 15 run attempts.

4.2. Single model

Table 1 shows the test accuracy with incorrect training, validation, and testing cases when adopting RGB video frames first and depth video frames second for the single proposed models where testing, validation, and training of the proposed models were done on our data for ArSL. Depending on the total number of incorrect cases in testing, validation, and training and for RGB video frames and depth video frames, the results of model 1 are the best compared to the rest of the models where the total incorrect cases are 17. Model 1 of the proposed four deep neural networks is the best model, which implements feature extraction through pre-trained models.

Table 1. The test accuracy with incorrect cases in training, validation, and testing when adopting RGB video frames first and depth video frames secondly for the single proposed models

	Model	Batch Size	Channels	Training Time	Incorrect Training	Incorrect Validation	Incorrect Test	Test Accuracy%	Total Incorrect
RGB	Model 1: ResNet50-LSTM	4	3	8h2min	0	2	2	99.62	4
	Model 2: Conv2D-LSTM	4	3	7h38min	68	20	17	96.76	105
	Model 3: Conv3D-LSTM	8	3	7h34min	192	34	36	93.14	262
	Model 4: Conv3D	8	3	6h57min	491	73	66	87.43	630
DEPTH	Model 1: ResNet50-LSTM	4	3	11h12min	4	8	6	98.86	18
	Model 2: Conv2D-LSTM	4	3	10h06min	65	17	14	97.33	96
	Model 3: Conv3D-LSTM	8	1	7h	801	86	83	84.19	970
	Model 4: Conv3D	8	1	9h50min	56	36	24	95.43	116

4.3. Multi-model

The focus was on model 1 using the pre-trained models with two different RNN as: ResNet50-LSTM, DenseNet121-LSTM, ResNet50-GRU, MobileNet-LSTM and VGG16-LSTM, where they were trained on our data for ArSL for RGB video frames and depth video frames as a single model and then multi-model. In the case of the multi-model, the same fusion method was adopted when using each of the five pre-trained models mentioned, which includes fusion before the last layer of the architecture, i.e. feature level fusion type Table 2 shows the test accuracy with incorrect cases in training, validation, and testing when adopting RGB video frames first and depth video frames secondly for the single and multi-models in the form of model 1. It can be seen from Table 2 that depending on total incorrect; in general, there is a convergence between the performance of the models, when they are used as a single model for RGB video frames or depth video frames, and also as a multi-model. It also shows that the best multi-model is the ResNet50-LSTM, where the sum of the total incorrect cases is 5 cases.

Table 2. The test accuracy with incorrect cases in training, validation, and testing when adopting RGB video frames first and depth video frames secondly for the single and multi-models in the form of model 1

	Model	Batch size	Channels	Training time	Incorrect train	Incorrect validation	Incorrect test	Test accuracy %	Total incorrect
RGB	ResNet50-LSTM	4	3	8h2min	0	2	2	99.62	4
	DenseNet121-LSTM	4	3	10h1min	0	2	3	99.43	5
	ResNet50-GRU	4	3	7h59min	0	3	3	99.43	6
	MobileNet-LSTM	8	3	7h44min	3	3	1	99.81	7
	VGG16-LSTM	8	3	8h43min	19	6	3	99.43	28
DEPTH	ResNet50-LSTM	4	3	11h12min	4	8	6	98.86	18
	DenseNet121-LSTM	4	3	9h50min	1	7	4	99.24	12
	ResNet50-GRU	4	3	9h35min	1	6	4	99.24	11
	MobileNet-LSTM	8	3	9h41min	1	3	7	98.67	11
	VGG16-LSTM	8	3	9h37min	43	17	12	97.71	72
MULTI-MODEL	ResNet50-LSTM	2	3	23h3min	0	3	2	99.62	5
	DenseNet121-LSTM	2	3	19h	1	6	0	100.00	7
	ResNet50-GRU	2	3	19h40min	0	3	3	99.43	6
	MobileNet-LSTM	4	3	21h43min	2	7	7	98.67	16
	VGG16-LSTM	4	3	23h2min	0	8	3	99.43	11

4.4. Evaluation of multi-model fusion

The type of multi-model fusion was evaluated with the approved Model 1(ResNet50-LSTM). Several types of multi-model fusion techniques we tested. Figure 4 shows the multi-model architecture of

types data level and feature level (A-H). We created a new multi-model G with the same multi-model H architecture and replaced the concatenated layer with the maximum layer. We then created a new multi-model I with the same multi-model H architecture and replaced the LSTM with a bidirectional LSTM (BLSTM). The performance of the model can be improved by using bidirectional LSTMs in sequence classification problems. Two LSTM layers are trained on the input sequences in bi-directional LSTM, which are an extension of traditional LSTMs, to offer extra information from the input and deliver quicker results. The first LSTM used forward temporal information to train on the input sequence, whereas the second used reverse temporal information to train on the reversed copy of the input sequence. To represent the sequence's bi-directional dependence, the forward and backward outputs will be concatenated. We then created a new multi-model J with the same architecture of multi-model H and added normalization layers for two models before concatenated layer. We normalized the values of both feature maps in the same range because the networks of both models create separate feature maps with different range values. After normalization, we concatenate layers to combine the values of the feature maps in order to increase the quality of the produced features. The architecture of multi-model K as it's for H with Bi-directional LSTM and normalization. Table 3 shows the test accuracy with incorrect cases in training, validation, and testing as well as the Inference on GPU and CPU for ResNet50 multi-models. The ResNet50 multi-models included data level fusion, several Feature level fusions, and decision level fusion. It is clear from Table 3 that the accuracy is not less than 99% for all the approved types except for data level multi-model fusion. When comparing the types of fusion at the data level, feature level or decision level, the best is the decision level. The best type for fusion at the feature level is the multi-model K, based on the total incorrect; where the total incorrect is zero and test accuracy is 100%. As for the number of FPS, Table 3 shows that the greatest FPS is when the fusion is at the data level, where the use of a single model is Model 1. The least FPS is when the fusion is at the decision level where two separate models are used. But when the fusion is at the features level, the FPS between the two cases where the multi-model is greater than a single model and less than two separate single models.

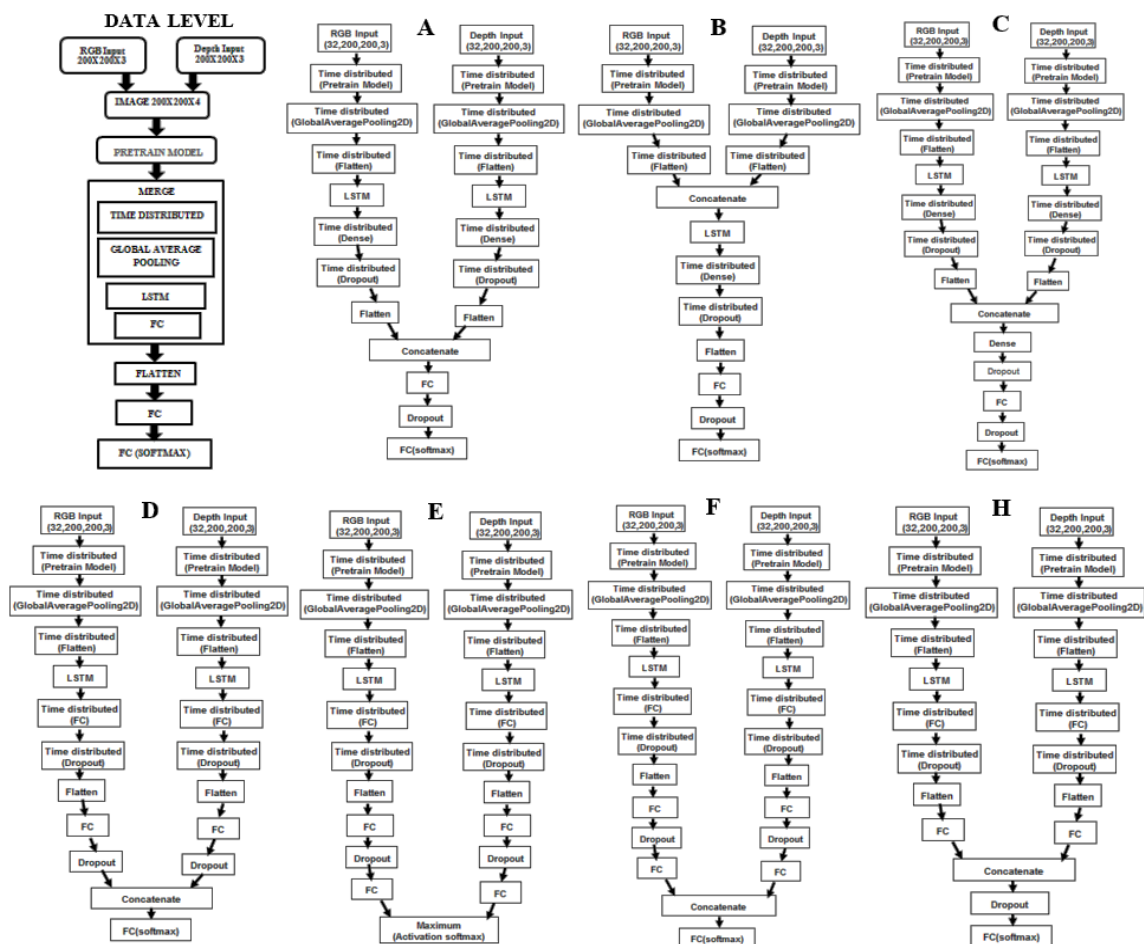


Figure 4. Different methods of fusion multiple models using the mode1/ResNet50-LSTM

Table 3. The test accuracy with incorrect cases in training, validation, and testing as well as the inference on GPU and CPU for ResNet50 multi-models

Multi-Model	Batch Size	channels	Training Time	Incorrect Training	Incorrect Validation	Incorrect Test	Accuracy %	Incorrect Total	Inference on GPU			Inference on CPU			
									FPS	GPU Usage %	CPU Usage %	FPS	GPU Usage %	CPU Usage %	
Data level	4	4	26h40min	0	26	30	94.29	56	200	87	20	26	0	100	
Feature level	A	2	3	24h33min	0	4	2	99.62	6	128	95	23	13	0	100
	B	2	3	25h16min	1	3	2	99.62	6	133	100	21	11	0	100
	C	2	3	28h23min	0	2	2	99.62	4	128	94	23	14	0	100
	D	2	3	27h25min	3	4	1	99.81	8	124	96	23	14	0	100
	E	2	3	23h57min	1	4	3	99.43	8	129	96	19	14	0	100
	F	2	3	29h42min	1	5	1	99.81	7	128	98	18	14	0	100
	G	2	3	23h42min	2	3	3	99.43	8	128	93	19	14	0	100
	H	2	3	23h3min	0	3	2	99.62	5	125	98	24	13	0	100
	I	2	3	23h12min	0	4	0	100.00	4	105	81	26	10	0	100
	J	2	3	26h30min	0	1	1	99.81	2	122	96	20	14	0	100
	K	2	3	23h37min	0	0	0	100	0	104	81	27	10	0	100
Decision level				0	0	0	100	0	96	92	22	9	0	100	

ResNet50-BiLSTM-Normalization validation and training loss, as well as validation and training accuracy, are shown in Figure 5. During the training and validation phases, the accuracy continues to increase as the loss rate decreases. The confusion matrix in Figure 6 shows the multi-model neural network evaluation results for the testing and training networks of ResNet50-BiLSTM-Normalization. The model is utilized for data augmentation approaches in ArSL video classification, as seen in the table of 21-class confusion matrix. All correction categories are grouped on a square matrix's diagonal. Columns represent the actual classes, and rows represent the classifier's predictions. Table 4 shows the Comparison for the ArSL between previous works [7], [9], [13], [18], [27]-[30] and this work. From this table, both proposed methods for a single model and multi-model are better than the models presented in the previous studies referred to in the table.

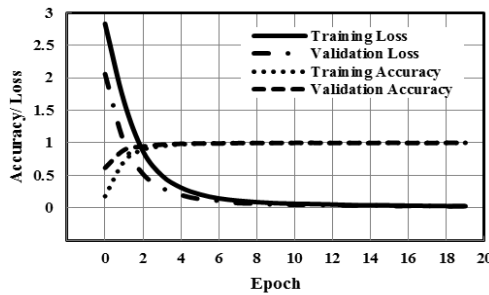


Figure 5. The training and validation accuracy in addition to the training and validation loss of ResNet50BiLSTM+normalization

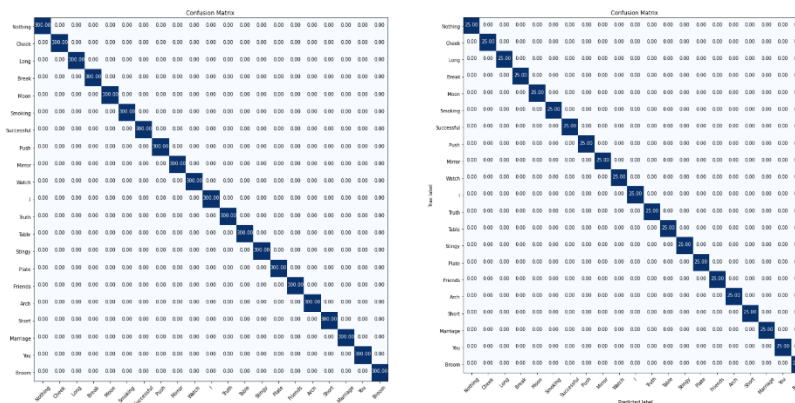


Figure 6. Training and testing confusion matrix of multi-model ResNet50BiLSTM+ normalize

Table 4. Comparison for ArSL of previous works and this work

Paper no.	Method	Signs/signer	Accuracy
[7]	Single model: 3D-CNN	25/5	98.00%
[9]	Single model: Dynamic time warping DTW & Kinect censor	30/-	97.58%
[13]	Single model: DeepLabv3+CSOM+BiLSTMNet	23/3	89.59%
[18]	Single model: Inception v3 CNN-LSTM	9/2	90.00%
[27]	Single model: Euclidean distance	30/-	97.00%
[28]	Single model: KNN, SVM, MLP	23/3	99.00%
[29]	Single model: 3D-CNN		96.69%
	Multi-model: 3D-CNN	40/4	98.12%
[30]	Single model: Hidden Markov Model	20/-	82.20%
This work	Single model: ResNet50-LSTM(RGB)		99.62%
This work	Single model: ResNet50-GRU(Depth)	21/55	99.24%
This work	multi-model: ResNet50-BiLSTM-Normalization		100.00%

5. CONCLUSION

Various models have been investigated for dynamic hand gesture recognition using RGB and depth information, through their analysis and discussion results, the following was concluded: The research prepared recorded and collected data including 7,350 RGB videos and 7350 depth videos as ArSL datasets. Model 1 of the proposed four deep neural networks is the best model, which implements feature extraction through pre-trained models. Among the pre-trained models with two different RNN: ResNet50-LSTM, DenseNet121-LSTM, ResNet50-GRU, MobileNet-LSTM and VGG16-LSTM, the best multi-model is the ResNet50-LSTM, with the same fusion method. The test accuracy is greater than 99% for all the approved multi-model fusion types for the feature level. The research presented the optimal multi-model ResNet50-BiLSTM-Normalization with 100% test efficiency and without incorrect training, validation and test.




REFERENCES

- [1] P. Kumar, H. Gauba, P. Pratim Roy, and D. Prosad Dogra, "A multimodal framework for sensor-based sign language recognition," *Neurocomputing*, vol. 259, pp. 21-38, Oct. 2017, doi: 10.1016/j.neucom.2016.08.132.
- [2] E. K. Kumar, P. V. V. Kishore, M. T. K. Kumar, and D. A. Kumar, "3D sign language recognition with joint distance and angular coded color topographical descriptor on a 2_Stream CNN," *Neurocomputing*, vol. 372, pp. 40-54, Jan. 2020, doi: 10.1016/j.neucom.2019.09.059.
- [3] A. V. Nair and V. Bindu, "A Review On Indian Sign Language Recognition," *International journal of computer applications*, vol. 73, no. 22, pp. 33-38, 2013, doi: 10.5120/13037-0260.
- [4] H. Vo, V. H. Pham, and B. T. Nguyen, "Deep Learning for Vietnamese Sign Language Recognition in Video Sequence," *International Journal of Machine Learning and Computing*, vol. 9, no. 4, pp. 440-445, 2019, doi: 10.18178/ijmlc.2019.9.4.823.
- [5] E. Lachat, H. Macher, T. Landes, and P. Grussenmeyer, "Assessment and Calibration of A RGB-D Camera (Kinect V2 Sensor) Towards A Potential Use For Close-Range 3D Modeling," *Remote Sensing*, vol. 7, no. 10, pp. 13070-13097, 2015, doi: 10.3390/rs71013070.
- [6] S. Paul, S. Basu, and M. Nasipuri, "Microsoft Kinect In Gesture Recognition: A Short Review," *Int. J. Control Theory Appl.*, vol. 8, no. 5, pp. 2071-2076, 2015.
- [7] M. ElBadawy, A. S. Elons, H. A. Shedeed, and M. F. Tolba, "Arabic Sign Language Recognition With 3d Convolutional Neural Networks," In *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, IEEE, pp. 66-71, 2017, doi: 10.1109/INTELICIS.2017.8260028.
- [8] S. Masood, A. Srivastava, H. C. Thuwal, and M. Ahmad, "Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN," In *Intelligent Engineering Informatics*, Springer, Singapore, pp. 623-632, 2018, doi: 10.1007/978-981-10-7566-7_63.
- [9] S. Abdel, A. Abdel-Rabouh, F. A. Elmisery, A. M. Brisha, and A. H. Khalil, "Arabic Sign Language Recognition Using Kinect Sensor," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 15, no. 2, pp. 57-67, 2018, doi: 10.19026/rjaset.15.5292.
- [10] Y. Liao, P. Xiong, W. Min, W. Min, and J. Lu, "Dynamic Sign Language Recognition Based on Video Sequence With BLSTM-3D Residual Networks," *IEEE Access*, vol. 7, pp. 38044-38054, 2019, doi: 10.1109/access.2019.2904749.
- [11] E. Zhang, B. Xue, F. Cao, J. Duan, G. Lin, and Y. Lei, "Fusion Of 2D-CNN and 3D Densenet For Dynamic Gesture Recognition," *Electronics*, vol. 8, no. 12, p. 1511, 2019, doi: 10.3390/electronics8121511.
- [12] W. Zhang, Wenjin, and J. Wang, "Dynamic hand gesture recognition based on 3D convolutional neural network models," In *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, IEEE, pp. 224-229, 2019, doi: 10.1109/ICNSC.2019.8743159.
- [13] S. Aly and W. Aly, "DeepArSLR: A Novel Signer-Independent Deep Learning Framework for Isolated Arabic Sign Language Gestures Recognition," *IEEE Access*, vol. 8, pp. 199-212, 2020, doi: 10.1109/ACCESS.2020.2990699.
- [14] D. S. Tran, N. H. Ho, H. J. Yang, E. T. Baek, S. H. Kim, and G. Lee, "Real-Time Hand Gesture Spotting and Recognition Using RGB-D Camera And 3D Convolutional Neural Network," *Applied Sciences*, vol. 10, no. 2, p. 722, 2020, doi: 10.3390/app10020722.
- [15] D. Sarma, V. Kavyasree, and M. K. Bhuyan, "Two-Stream Fusion Model For Dynamic Hand Gesture Recognition Using 3d-Cnn And 2d-Cnn Optical Flow Guided Motion Template," *arXiv preprint arXiv:2007.08847*, 2020.
- [16] R. Rastgoo, K. Kiani, and S. Escalera, "Video-Based Isolated Hand Sign Language Recognition Using A Deep Cascaded Model," *Multimedia Tools and Applications*, vol. 79, pp. 22965-22987, 2020, doi: 10.1007/s11042-020-09048-5.
- [17] E. K. Elsayed and D. R. Fathy, "Semantic Deep Learning to Translate Dynamic Sign Language," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, pp. 316-325, 2021, doi: 10.22266/ijies.2021.0228.30.
- [18] Elhagry and R. Gla, "Egyptian Sign Language Recognition Using CNN and LSTM," *arXiv preprint arXiv: 2107.13647*, 2021.




- [19] Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1-48, 2019, doi: 10.1186/s40537-019-0197-0.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778, 2016, doi: 10.1109/CVPR.2016.90.
- [21] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700-4708, 2017, doi: 10.1109/CVPR.2017.243.
- [22] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv Preprint arXiv: 1409.1556*, 2014.
- [23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "Mobilenetv2: Inverted Residuals and Linear Bottlenecks," in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510-4520, 2018, doi: 10.1109/CVPR.2018.00474.
- [24] M. Gao, J. Jiang, G. Zou, V. John, and Z. Liu, "RGB-D-Based Object Recognition Using Multimodal Convolutional Neural Networks: A Survey," *IEEE access*, vol. 7, pp. 43110-43136, 2019, doi: 10.1109/ACCESS.2019.2907071.
- [25] F. Farahnakian and J. Heikkonen, "Deep Learning Based Multi-Modal Fusion Architectures for Maritime Vessel Detection," *Remote Sensing*, vol. 12, no. 16, p. 2509, 2020, doi: 10.3390/rs12162509.
- [26] Y. Gao *et al.*, "Estimating Gpu Memory Consumption of Deep Learning Models," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 1342-1352, 2020, doi: 10.1145/3368089.3417050.
- [27] N. B. Ibrahim, M. M. Selim, and H. H. Zayed, "An Automatic Arabic Sign Language Recognition System (Arslrs)," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 4, pp. 470-477, 2018, doi: 10.1016/j.jksuci.2017.09.007.
- [28] H. Luqman and S. A. Mahmoud, "Transform-Based Arabic Sign Language Recognition," *Procedia Computer Science*, vol. 117, pp. 2-9, 2017, doi: 10.1016/j.procs.2017.10.087.
- [29] M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, M. A. Bencherif, and M. A. Mekhtiche, "Hand Gesture Recognition for Sign Language Using 3DCNN," *IEEE Access*, vol. 8, pp. 79491-79509, 2020, doi: 10.1109/ACCESS.2020.2990434.
- [30] A. Elsayed and H. Hamdy, "Arabic Sign Language (Arsl) Recognition System Using HMM," *International Journal of Advanced Computer Science and Applications*, vol. 2, no. 11, 2011, doi: 10.14569/IJACSA.2011.021108.

BIOGRAPHIES OF AUTHORS






Mohammad Haqqi Ismail    received the BSc and MSc degree in Computer Engineering in 2009 and 2017 from University of Mosul, IRAQ. He is work as assistant lecturer in Technical Computer Engineering, Al-Hadba University College, Mosul, IRAQ. Currently, he is PhD student at research stage in Computer Engineering Department University of Mosul, IRAQ. He researches interests include image processing, deep learning and parallel processing. He can be contacted at email: mohammadhaqqi@gmail.com.



Prof. Dr. Shefa A. Dawwd    is a professor of computer engineering at the Computer Engineering Department-University of Mosul. He received the B.Sc in Communication Engineering, the M.Sc and the Ph.D in Computer Engineering. He has authored about 40 international journal, conference papers and one-chapter book. His research focus is on the processing acceleration of 1D, 2D and 3D signals, real time applications, deep learning, Convolutional Neural Networks, and heterogeneous computing. He is a regular reviewer of IEE, Elsevier and other Scopus journals. He can be contacted at email: sheaf.dawwd@uomosul.edu.iq.



Dr. Fakhruddin H. Ali    is assistant professor at the computer engineering Department-University of Mosul. He received B.Sc in Electronic and Communication Engineering-Department of Electrical Engineering-University of Mosul. He received P.G. Diploma and M.Sc from the same Department at 1977-1979. He graduated from university of Bradford-U.K. with a PhD degree at 1989. He has more than 30 scientific papers in journals and conferences. He supervised more than 25 postgraduate M.Sc and PhD. Theses and dissertations. His field of interest is 3D computer graphics and real time systems. He can be contacted at email: fhazaa@uomosul.edu.iq.