Static hand gesture recognition of Arabic sign language by using deep CNNs

Mohammad H. Ismail, Shefa A. Dawwd, Fakhradeen H. Ali
Department of Computer Engineering, College of Engineering, University of Mosul, Mosul, Iraq

Article Info

Article history:

Received Jun 17, 2021
Revised Jul 31, 2021
Accepted Aug 4, 2021

Keywords:

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

ABSTRACT

An Arabic sign language recognition using two concatenated deep convolution neural network models DenseNet121 & VGG16 is presented. The pre-trained models are fed with images, and then the system can automatically recognize the Arabic sign language. To evaluate the performance of concatenated two models in the Arabic sign language recognition, the red-green-blue (RGB) images for various static signs are collected in a dataset. The dataset comprises 220,000 images for 44 categories: 32 letters, 11 numbers (0:10), and 1 for none. For each of the static signs, there are 5000 images collected from different volunteers. The pre-trained models were used and trained on prepared Arabic sign language data. These models were used after some modification. Also, an attempt has been made to adopt two models from the previously trained models, where they are trained in parallel deep feature extractions. Then they are combined and prepared for the classification stage. The results demonstrate the comparison between the performance of the single model and multi-model. It appears that most of the multi-model is better in feature extraction and classification than the single models. And also show that when depending on the total number of incorrect recognize sign image in training, validation and testing dataset, the best convolutional neural networks (CNN) model in feature extraction and classification Arabic sign language is the DenseNet121 for a single model using and DenseNet121 & VGG16 for multi-model using.

Corresponding Author:

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

1. INTRODUCTION

Sign language is thought to be the only means for normal people, hearing-impaired, and deaf to communicate. People use non-verbal speech in the form of sign language signals to express their thoughts and feelings. In sign language, there are two types of gestures: static and dynamic. [1]. Arabic Sign Language ArSL has many nation varieties and dialects. It varies from one nation to another, even often inside the same country. Despite this, the alphabet and numbers in the Arabic language are standardized in sign language [2]. Sign languages need an intelligent device that can convert them from one sign language to another using natural language. Without an interpreter, it's difficult for most people who aren't interested in sign language to communicate. These problems necessitate the use of automatic sign language translation programs.

Non-manual and manual signs are the two major components that make up sign languages. Body motion and facial expressions are represented by the non-manual. Hand location, orientation, shape, and trajectory are the manual signals. Most works, however, concentrate on manual signs because they provide the most important information, non-manual signs, on the other hand, assist signers in explaining and emphasizing the value of manual signs [3], [4]. In this work, the manual sign is investigated.

Sign language detection is achieved in two approaches: the first approach depends on vision-based, while the second approach is based on sensor-based [5], [6]. The vision-based approach captures the hand gesture with the camera in the form of static or sequential images without the use of gloves or sensors. This approach is most appropriate for the real daily life of the deaf and mute, although there are many obstacles such as lighting conditions, skin colour, background differences, in addition to the properties and settings of the camera [5]. The sensor-based approach involves wearing gloves, which contain sensors intended to express sign language. These gloves have the characteristic of being unaffected by the obstacles that the vision-based approach faces. However, it is not appropriate to wear it most of the time [5].

The main goal of the presented research is to prepare data related to the Arabic sign language, amounting to 220 thousand colour images. Then, using pre-trained models fed with images, set up a system that can automatically recognize the Arabic sign language, which includes 44 categories: 32 letters, numbers from 0 to 10, and one for none. Also, an attempt to evaluate the performance of concatenating two models in the Arabic sign language recognition is presented.

2. RELATED WORKS

Deep learning is widely used in many areas. Convolutional neural networks are a form of deep neural network that is widely used for image analysis. There are various architectures available for convolutional neural networks (CNNs). CNNs are giving the best and most accurate results when solving real-world problems. One of its applications is image classification, which is the process of capturing an image as an input and producing the image’s class. A critically important good prediction can be obtained through CNNs role in reducing images to a form that is easy to process without losing features.

Many researchers have used different methods to identify sign language in general or Arabic, and some of them will be presented. In [7], a method for recognizing ArSL numbers and letters is suggested. With a real dataset of 5839 images of 28 characters and 2030 images of numbers (from 0 to 10), this system is based on CNN. The proposed system has a recognition rate of 90.02%.

Using a fine-tuned VGG19 model, Crepso *et al.* [8] proposes an red-green-blue (RGB) and RGB-D static gesture recognition system. The fine-tuned VGG19 model uses a feature concatenate layer of RGB and RGB-D images to increase the neural network's accuracy. The proposed model tested an American sign language (ASL) Recognition dataset achieved a 94.8% recognition rate.

Dadashzadeh *et al.* [9] suggested a two-stage fusion network based on CNN architecture for hand gesture recognition. In the first stage of the network, they proposed hand segmentation architecture. When there is a similarity between skin colour and background colour, the hand segmentation model performed well in difficult conditions, according to their data. They designed a two-stream CNN for the network's second level until classification, it can learn to merge feature representations from both the RGB image and its segmentation map. Their system runs at a frame rate of 23 ms per frame.

A deep learning-based method for ArSL recognition was suggested in [10]. Deep features are selected by processing input images with various layers. Finally, the SoftMax function is used to divide target classes into categories and compute a normalized probability score for each. With a score of 99.52%, the suggested system based on residual network ResNet101 obtained the greatest accuracy. Elsayed and Fathy [11] trained and tested Deep CNN architecture on an Arabic sign language dataset. Their experimental results show that the training set's classification accuracy was 98.6%, while the testing sets was 94.31%, according to the collected dataset.

Althagafi *et al.* [12] used a CNN model by taking grayscale images as input to a system that automatically recognizes 28 letters for Arabic Sign Language recognition, they achieved 92.9% of recognition accuracy on 10810 tested images. Latif *et al.* [13] suggested a system that recognizes the Arabic alphabet's signs in real-time. A database of more than 50000 images was used to train and test the Deep CNN architectures. Several trials are carried out to determine the highest recognition rates by changing CNN architectural design parameters. Three convolutional layers, three pooling layers, and a fully connected layer make up the proposed deep CNN architecture. The accuracy of the experimental results is 97.6%.

The accuracy of recognizing 32 hand gestures from the Arabic sign language is improved using transfer learning and fine-tuning deep convolutional neural networks (VGG16, ResNet152) [14]. The implementation of the presented model was accomplished by reducing the size of the training dataset while increasing accuracy. The networks were fed by images of various Arabic Sign Language data and were able to achieve an accuracy of approximately 99%. The convolutional neural network (CNN) and a dataset of 20,000 sign images of 10 static digits were used in research [15] to build the BSL digits recognition system. The proposed CNN model was compared to a number of other sign language models. The proposed CNN model's architecture was close to that of the VGGNet, but it only had six convolutional layers instead of the VGGNet's minimum of 13. The training accuracy of the proposed system is 97.62%.

The system is a re-training VGG system [16] for real-time ASL fingerspelling recognition with CNNs networks to classify a total of 26 alphabets, as well as two classes for space and delete. The system had a training set accuracy of 98.53% and a validation set accuracy of 98.84%. CNN used the proposed system [17] to recognize Arabic hand sign-based letters and translate them into Arabic speech. This system has a 90% accuracy rate in recognizing Arabic sign letters. Tasmere *et al.* [18] introduced a system to recognize hand gestures in real-time. Hand segmentation in the YCbCr colour space was used for gesture identification, followed by the suggested CNN model. Three convolution layers, two max-pooling layers, and two fully connected layers represent the proposed CNN model. For 11 gestures from depth images, this proposed technique provided an accuracy of 94.61%. A dataset containing 1320 sample images was used. In the current study, there are several attempts to develop both the single model and the multi-models to increase the performance and accuracy of the Arabic sign language recognition. In addition, this study was distinguished by the following:

– A large-sized colored dataset was prepared for the Arabic sign language due to the inability to access such data, by many researchers who deal with Arabic sign language recognition.
– According to the previous researches and using the multi-model's method, there are different input data for each model as colour and depth images. While in this study, the same input colour images were used for each model.

The CNN models generate different lengths of feature maps with different ranges of values. When using multiple models, before merging the two models features, we normalized the values of these feature maps in the same range.

3. EXPERIMENTAL METHODOLOGY

3.1. Dataset

The three-channel RGB images are received from the camera. The RGB images for various static signs are collected in this dataset. The dataset comprises 220,000 images for 44 categories: 32 letters as shown in Figure 1 to express all the Arabic sign language (ArSL) vocabulary, 11 numbers (0:10), and 1 for none. For each of the static signs, there are 5000 images collected from 10 different volunteers. The dataset divided into three groups training, validation, and testing, where 80% (176,000 images) of the data were used for training, 10% (22,000 images) of the data were used for validation, and 10% (22,000 images) of the data were used for testing. The dataset also included several cases of diverse lighting conditions and backgrounds; it included changing the distance between a user and the camera, as shown in Figure 2.

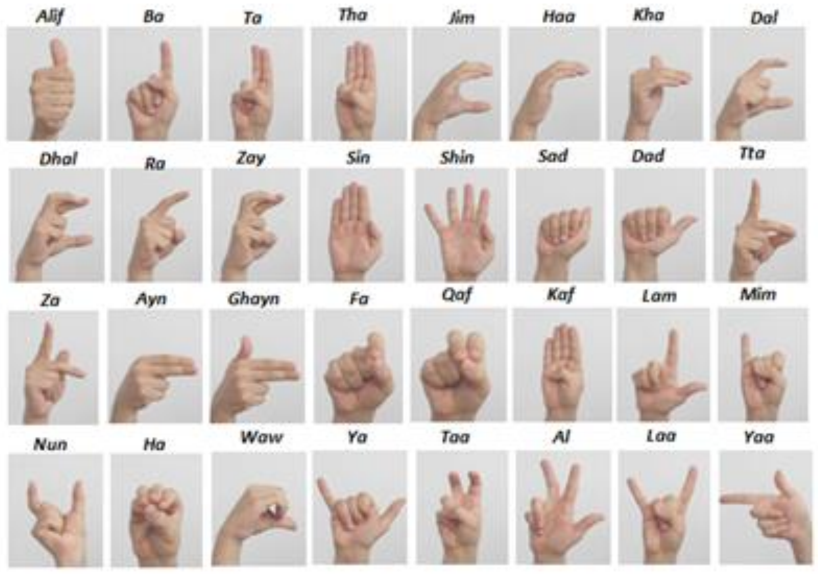


Figure 1. Arabic alphabet signs are type of static gestures and are performed using a single hand

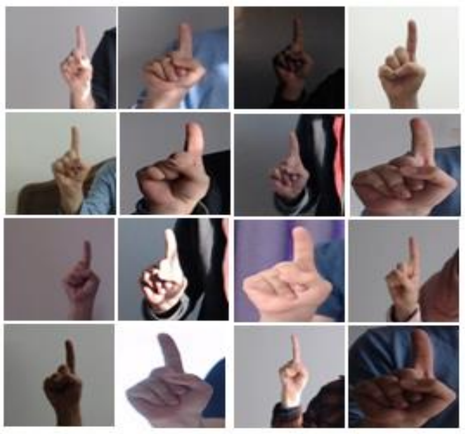


Figure 2. A set of images of the letter Ba from the dataset

3.2. Data preprocessing

Sign images were preprocessed by resizing and normalizing the image. The image is then resized to 100x100. This size is chosen as a tradeoff between accuracy and execution time. These images are then normalized to change the range of pixel intensity values, resulting in a mean value of 0 and a variance of 1.

3.3. Data augmentation

Usually, for these very powerful deep neural networks, deep learning is associated with millions of images. The disadvantage of the limited training image set is that the neural network may remember our training data and can predict the performance of the training set well, but the verification accuracy is poor. For solving the dataset problem, data augmentation was applied to prevent overfitting and improve model generalization ability [19]. The study uses online data augmentation. There are various data augmentation techniques used for static sign language to prevent model overfitting and enhance learning capability: Normalization image, brightness range (0.4-1.2), zoom range (1.0, 1.2), height shift range (10%), width shift range (10%), rotation range (±10°). The augmentation of data for the dynamic sign was done by applying rotation ± (5°-10°), translation transformation ± (4-8%) and change the brightness ± (8-28%) and sharpen the image, added noise salt and paper and blurring images with filters gaussian, median, averaging and morphological operation erosion and dilation of the dataset. We also flipped the images horizontally to include left or right-handed sign language. The training set is increased about 48 times through these operations. Inside the mini-batch fed into the model, all of these operations are applied at random.

3.4. Pre-train models

To take advantage of Transfer learning by using pre-trained models. ImageNet is a research project that aims to create a massive image database. Models such as the DenseNet121 [20], VGG16 [21], NasnetMobile [22], Xception [23], MobileNetV2 [24], EfficientB0 [25], InceptionV3 [26] and ResNet50 [27] were trained on various classes of images. These models were created from scratch and trained on millions of images containing thousands of objects using high-quality GPUs. The model has learned a good representation of low-level features such as spatial, edges, rotation, illumination, and shapes since it was trained on a large dataset. These features may be exchanged to facilitate transfer learning and extract features from new images across several computer vision problems. The previously tested model should also be able to extract specific features from these new images based on the concepts of transfer learning, even though the new images are from entirely different groups than the source dataset. This is to benefit from these models in extracting features and classifying images, which are different from what they were trained on. Therefore, this requires changing the last layers responsible for classification from these models with different other layers to match the number of objects to be classified. Then training on the new image data until the desired precision in recognition is obtained. This method is considered the best one in obtaining the required accuracy in recognizing from adopting untrained models.

3.5. Proposed method

3.5.1. Single model

Pre-trained models with trained weights are used on the ImageNet. These models (DenseNet121, VGG16, RESNet50, MobileNetV2, Xception, Efficient B0, NASNet Mobile, and InceptionV3) were used with some modifications. Each of these models includes two parts, the first for extracting features and the second for classification. The second part has been removed, and we have kept the feature extraction part. Then, a layer of global average pooling was added after the last layer in the feature extraction part. The global average pooling layer (GAP) is added to reduce the size of the feature map by converting it into a one-dimensional matrix while keeping vital information, where the size of a feature map with dimensions h×w×d is reduced to dimensions size to 1×1×d. GAP layers use the average of all h×w values to reduce each hw feature map to a single number. Attempts have also been made to add different layers for optimum classification. The best accuracy was when adding one layer of the dropout rate of 20% reduction in existing connections to prevent overfitting, in which the connections between the layers are randomly eliminated, the dropout layer is disabled in testing and validation mode. Then followed by a fully connected output layer (FC) of size 44, its unit’s number equal to class’s number, with a softmax activation function for classification. The following models were developed according to what was mentioned above: DenseNet121, VGG16, RESNet50, MobileNetV2, Xception, Efficient B0, NASNetMobile, and InceptionV3. Then each of them was trained on an Arabic sign language dataset to recognize Arabic sign language. And Figure 3 shows the general layout of architecture for each of these models.

3.5.2. Multi-model

An attempt has been made to adopt two models from the previously trained models referred to above, where they are trained in parallel deep feature extractors. Then they are combined and prepared for the classification stage. Figure 4 shows the architecture of a multi-model network, which consists of two branches. Each branch is a CNN model. DenseNet121 model and VGG16 model are used in the case shown in Figure 4. In this multi-model, our dataset's pre-processed input colour images size is 100x100 pixels, which represent the input image for two multi-model branches. From the input image, DenseNet produces a 3x3x1024 feature map on its last feature extractor layer, while VGG16 generates a 3x3x512 feature map on its last feature extractor layer. To reduce the size of the last layer feature map, we applied Global Average Pooling by taking the average of each feature map and extract important features. Since the networks of both models generate different feature maps with different range values, then we normalized the values of these feature maps in the same range by using the lambda layer. After the normalization, we combine these values of the features maps by concatenating layers to improve the quality of the created semantic features.

For both single model and multi-model during the training process, the data augmentation is one of the most popular methods for reducing overfitting. When the model is trained on the GPU, the data augmentation is performed in real-time on the CPU. Experiments are run on a single computer with an Intel Core i7-9750H Hexa-core CPU, 16GB SDRAM, and an NVIDIA GeForce RTX 2060 GPU with 6GB of memory. Python modules are used to implement the neural network models.

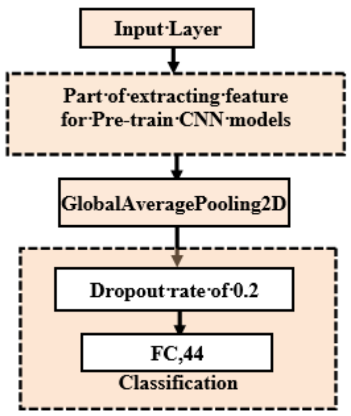


Figure 3. The architecture of the single model network

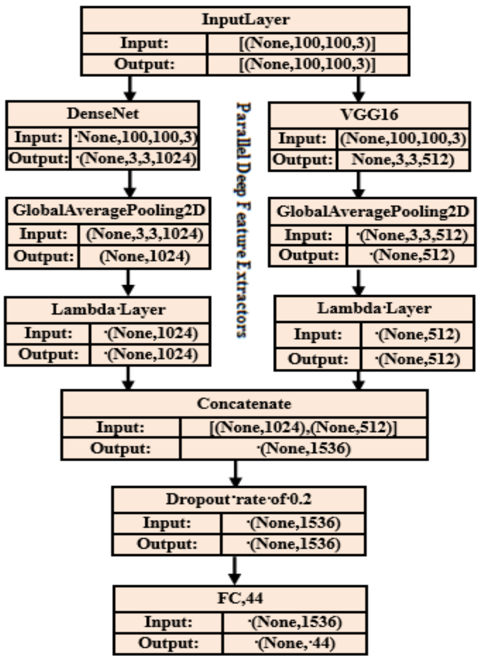


Figure 4. The architecture of the concatenated network

3.6. Evaluation of proposed method

To evaluate the performance of the best models proposed in this study by comparing them with other studies working on a standard data set. Since the proposed models were trained on the Arabic sign language data which we prepared, so the models proposed in this study were retrained on the American sign language (ASL) standard data, which Kaggle challenge developed. The ASL data includes 87,000 images divided into 29 categories: 26 letters for all ASL vocabulary and three letters for space, delete, and empty. The ASL data set was divided into three sets for training, validation and testing, 80% (69600 images) of the data were used for training, 10% (8700 images) of the data were used for validation, and 10% (8700 images) of the data used for testing. Thus, the model's performance in this study can be compared with previous studies that used the same ASL dataset.

3.7. General Workflow of the proposed method

The open-source Google MediaPipe technology is using to detect the hands. This platform allows using real-time computer vision technology, including hand detection, hand tracking. It was released in 2020. The Google MediaPipe technology provides detailed real-time finger tracking with multiple hands. The accuracy of the palm detection is 95%. MediaPipe uses two convolutional neural network models to detect the hand: palm detection and finger detection from a picture or video clip. This was used to define the hand region that would be extracted [28]. The sequence of frames captured by the camera is passed through a mediapipe framework hand detector to find the hand boundary. After that, the hand region is extracted and passed into the preprocessing stage to resize and normalise the hand region image. Then the hand region image passed into single or multi-CNN models for sign language recognition by feature extraction and classification. Figure 5 shows the overall architecture of the system for hand detection and sign language recognition.

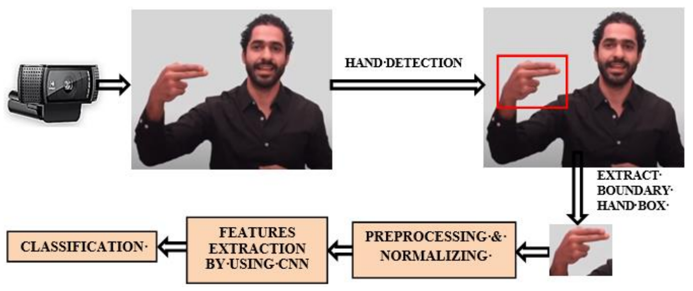

HAND DETECTION
EXTRACT BOUNDARY HAND BOX
PREPROCESSING & NORMALIZING
FEATURES EXTRACTION BY USING CNN
CLASSIFICATION

Figure 5. The general workflow of the proposed method for hand detection and gesture recognition

4. RESULTS AND DISCUSSION

The performance evaluation of our proposal has been carried out with the following metrics [29], widely used for this kind of task: Accuracy, Precision, Recall and F1-score. They are defined as follows:

$$Recall = \frac{true\ positives}{true\ positive+false\ negatives} \quad (1)$$

$$Precision = \frac{true\ positives}{true\ positive+true\ negatives} \quad (2)$$

$$Accuracy = \frac{true\ positives+true\ negatives}{Total\ instances\ classified} \quad (3)$$

$$F-Measure = \frac{2\times(Precision\times Recall)}{(Precision+Recall)} \quad (4)$$

False negative is a result under which the model forecasts the negative class wrongly. False positive is a result under which the model forecasts the positive class wrongly. True negative is a result under which the model forecasts the negative class accurately. True positive is a result under which the model forecasts the positive class accurately.

Table 1 compares the validation accuracy and test accuracy of a single model and multi-model with epochs equal to 5. Accuracy: is the ratio of the number of correct classifications to the total number of classifications. Incorrect recognize training, validation, and testing: are the number of misclassified images for training, validation, and testing respectively. The table revealed that both the validation accuracy and the test accuracy were at least 97% in most single or multimodal models. The accuracy ratio is high despite several incorrect recognize sign images, whether in testing or validation due to the large size of the data, where 22,000 images of the data were used for each of testing and validation. The table also shows the comparison between the performance of the single model and multi-model. It appears that the multi-model is better in feature extraction and classification than the single.

Table 1. Comparison of the validation accuracy and test accuracy for single and multi-models with Epochs=5

Model	Incorrect Recognize Training	Incorrect Recognize Validation	Validation Accuracy %	Incorrect Recognize Test	Test Accuracy %
DenseNet121	12	3	99.99	1	100
VGG16	9	6	99.97	5	99.98
DenseNet121 & VGG16	1	1	100	1	100
RESNet50	34	10	99.95	6	99.97
MobileNetV2	45	3	99.99	6	99.97
RESNet50 & MobileNetV2	23	5	99.98	5	99.98
Xception	110	15	99.93	15	99.93
Efficient B0	267	53	99.85	38	99.83
Xception&Efficient B0	106	16	99.93	17	99.92
NASNetMobile	2334	328	98.51	320	98.55
InceptionV3	3883	508	97.69	491	97.77
NASNetMobile & InceptionV3	3304	415	98.11	417	98.10
DenseNet121 & MobileNetV2	7	2	99.99	1	100
DenseNet121&RESNet50	11	2	99.99	2	99.99

Table 2 shows total parameters, FPS, training time, size of feature maps and total incorrect recognize sign image out of 220 thousand for different models. Total parameters: The parameters selected by the network during the training process are considered the network parameters. Their number determines the complexity of the network and the possibility of better learning, but this needs more images to train the network. Training Time: The time is taken to train the network. Size of feature maps: the size of last feature extraction layer. The frame per second (FPS) is the most common unit of time used in object detection. It indicates the maximum number of frames that the network will process in a second. total incorrect recognize: the number of misclassified images.

The top three best in feature extraction and classification models are the multi-models, DenseNet121 & VGG16, DenseNet121 & MobileNetV2, and DenseNet121 & RESNet50. It is based on the total number of Incorrect Recognize sign images in the training, validation and testing dataset. It is clear from the table that the training time for the multi-model is greater than the training time for the single models that compose it and less than the training time for both single models. It also shows that the FPS in the multi-model case is less than the single model and ranges between 66-96% of the FPS of single models. It also revealed that when the total parameters are increased, the FPS decreases. These are evident in the multi-model in which the total parameters are greater than that of the single model.

Table 2. Total parameters, FPS and total incorrect recognize for different deep CNN models

Model	Total Parameters*10^6	Training Time (hour)	Size of Feature Maps	FPS For Inference	Total Incorrect Recognize
DenseNet121	7.08	2.13	3*3*1024	24	16
VGG16	14.73	1.52	3*3*512	32	20
DenseNet121 & VGG1	21.81	3.17	1536	22	3
RESNet50	23.67	1.58	4*4*2048	28	50
MobileNetV2	2.31	1.17	4*4*1280	32	54
RESNet50 & MobileNetV2	25.99	2.28	3328	24	33
Xception	20.95	2.00	3*3*2048	28	140
Efficient B0	4.10	2.23	4*4*1280	24	358
Xception & Efficient B0	25.05	3.55	3328	19	139
NASNetMobile	4.31	3.68	4*4*1056	21	2692
InceptionV3	21.89	2.18	1*1*2048	23	4882
NASNetMobile & InceptionV3	26.3	4.77	3104	16	4136
DenseNet121 & RESNet50	30.76	3.22	3072	23	15
DenseNet121 &MobileNetV2	9.39	3.12	2304	21	10

Several single and multi models have been trained and tested. DenseNet121 and VGG16 network extracting deep features is better than other networks based on the method described above. Compared to the concatenation of DenseNet121 and VGG16 neural networks with other neural networks. The higher accuracy is obtained with the concatenated network. We were able to help the network learn the representation of both by concatenating the feature vectors of both networks, which accurately represented the image and produced a better accuracy of prediction.

The single models used in the proposed method were arranged on Tables 1 and 2 according to accuracy that depends on the total Incorrect Recognize. After that, the multi-models were used for every two single models in the sequence, meaning four multi-models. Then other options were added using the best single model with other single models outside the sequence. Figure 6 shows the training and validation accuracy in addition to the training and validation loss of multi-model DenseNet121 and VGG16. The accuracy continues to increase, and the loss rate decreases during the training and validation phases.

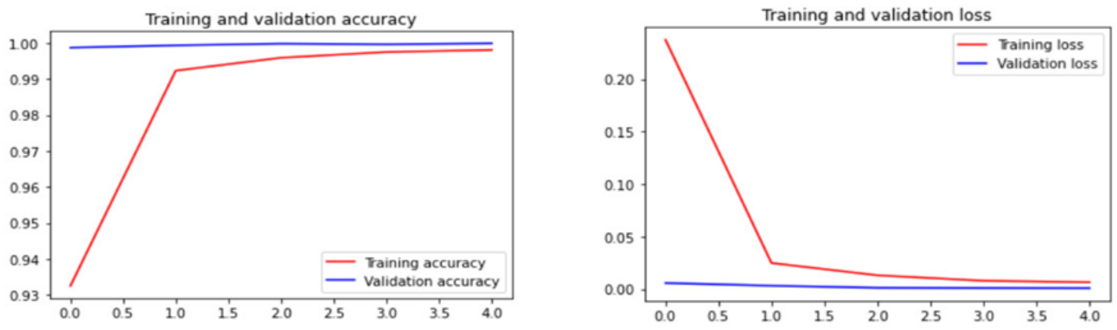


Figure 6. The training and validation accuracy in addition to the training and validation loss of multi-model DenseNet121 & VGG16

Table of 44-class confusion matrix the model is used for data augmentation techniques in ArSL image classification. Columns represent the true classes, and the classifier's predictions are represented by rows. All correction classifications are arranged in the diagonal of a square matrix. The results of the multi-model neural network evaluation of DenseNet121 and VGG16 are illustrated for the training and testing networks in the confusion matrix shown in Figures 7 and 8. Figure 9 show tabulation of precision, recall, f1-score, and support for each class of training network to recognize Arabic sign language with the task of the 44 class by multi-model DenseNet121 & VGG16.

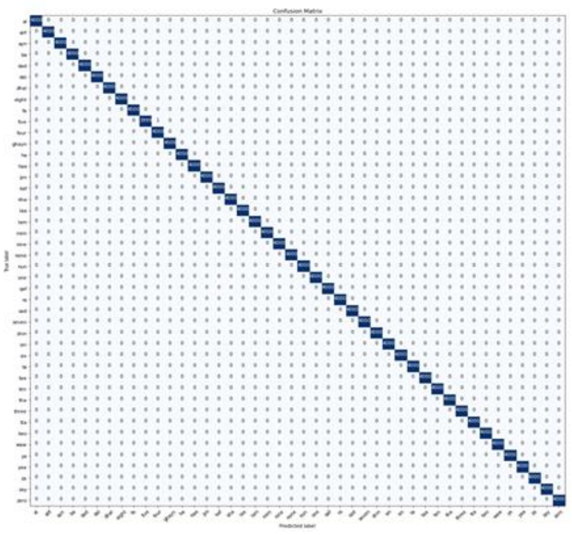


Figure 7. Training confusion matrix of multi-model DenseNet121 & VGG16

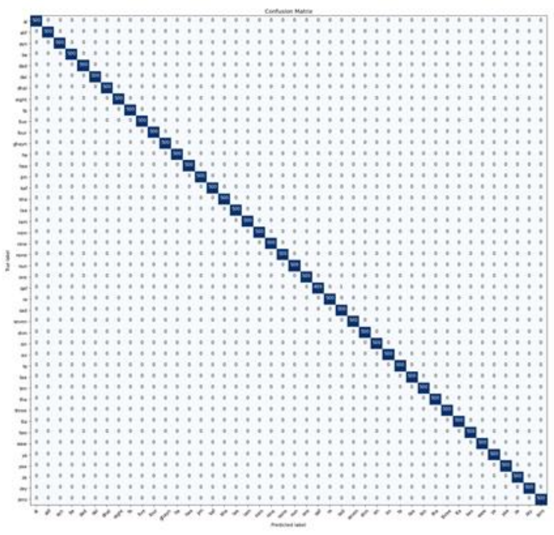
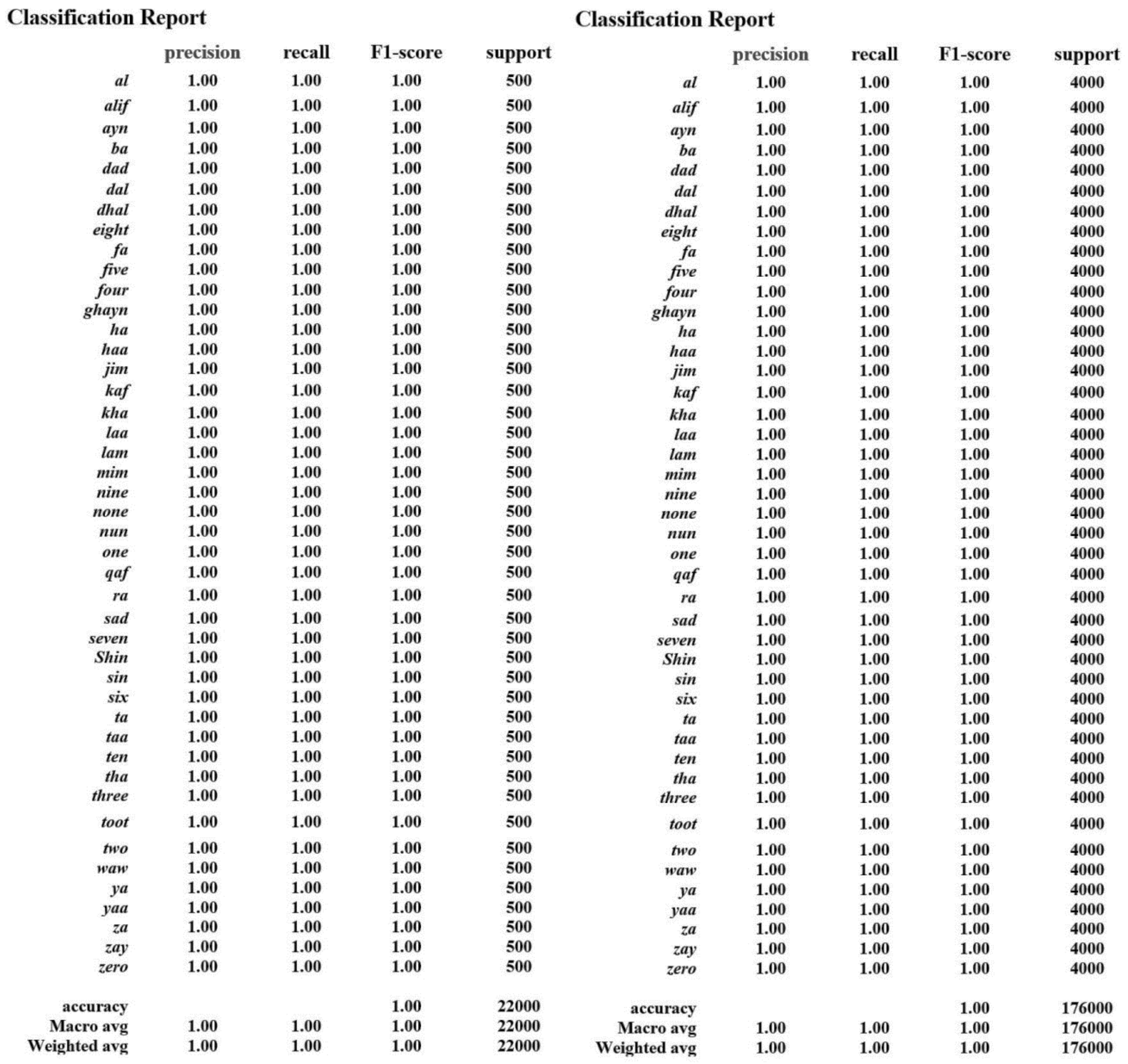


Figure 8. Testing confusion matrix of multi-model DenseNet121 & VGG16



Classification Report

	precision	recall	F1-score	support
al	1.00	1.00	1.00	500
alif	1.00	1.00	1.00	500
ayn	1.00	1.00	1.00	500
ba	1.00	1.00	1.00	500
dad	1.00	1.00	1.00	500
dal	1.00	1.00	1.00	500
dhal	1.00	1.00	1.00	500
eight	1.00	1.00	1.00	500
fa	1.00	1.00	1.00	500
five	1.00	1.00	1.00	500
four	1.00	1.00	1.00	500
ghayn	1.00	1.00	1.00	500
ha	1.00	1.00	1.00	500
haa	1.00	1.00	1.00	500
jim	1.00	1.00	1.00	500
kaf	1.00	1.00	1.00	500
kha	1.00	1.00	1.00	500
laa	1.00	1.00	1.00	500
lam	1.00	1.00	1.00	500
mim	1.00	1.00	1.00	500
nine	1.00	1.00	1.00	500
none	1.00	1.00	1.00	500
nun	1.00	1.00	1.00	500
one	1.00	1.00	1.00	500
qaf	1.00	1.00	1.00	500
ra	1.00	1.00	1.00	500
sad	1.00	1.00	1.00	500
seven	1.00	1.00	1.00	500
Shin	1.00	1.00	1.00	500
sin	1.00	1.00	1.00	500
six	1.00	1.00	1.00	500
ta	1.00	1.00	1.00	500
taa	1.00	1.00	1.00	500
ten	1.00	1.00	1.00	500
tha	1.00	1.00	1.00	500
three	1.00	1.00	1.00	500
toot	1.00	1.00	1.00	500
two	1.00	1.00	1.00	500
waw	1.00	1.00	1.00	500
ya	1.00	1.00	1.00	500
yaa	1.00	1.00	1.00	500
za	1.00	1.00	1.00	500
zay	1.00	1.00	1.00	500
zero	1.00	1.00	1.00	500
accuracy			1.00	22000
Macro avg	1.00	1.00	1.00	22000
Weighted avg	1.00	1.00	1.00	22000

Classification Report

	precision	recall	F1-score	support
al	1.00	1.00	1.00	4000
alif	1.00	1.00	1.00	4000
ayn	1.00	1.00	1.00	4000
ba	1.00	1.00	1.00	4000
dad	1.00	1.00	1.00	4000
dal	1.00	1.00	1.00	4000
dhal	1.00	1.00	1.00	4000
eight	1.00	1.00	1.00	4000
fa	1.00	1.00	1.00	4000
five	1.00	1.00	1.00	4000
four	1.00	1.00	1.00	4000
ghayn	1.00	1.00	1.00	4000
ha	1.00	1.00	1.00	4000
haa	1.00	1.00	1.00	4000
jim	1.00	1.00	1.00	4000
kaf	1.00	1.00	1.00	4000
kha	1.00	1.00	1.00	4000
laa	1.00	1.00	1.00	4000
lam	1.00	1.00	1.00	4000
mim	1.00	1.00	1.00	4000
nine	1.00	1.00	1.00	4000
none	1.00	1.00	1.00	4000
nun	1.00	1.00	1.00	4000
one	1.00	1.00	1.00	4000
qaf	1.00	1.00	1.00	4000
ra	1.00	1.00	1.00	4000
sad	1.00	1.00	1.00	4000
seven	1.00	1.00	1.00	4000
Shin	1.00	1.00	1.00	4000
sin	1.00	1.00	1.00	4000
six	1.00	1.00	1.00	4000
ta	1.00	1.00	1.00	4000
taa	1.00	1.00	1.00	4000
ten	1.00	1.00	1.00	4000
tha	1.00	1.00	1.00	4000
three	1.00	1.00	1.00	4000
toot	1.00	1.00	1.00	4000
two	1.00	1.00	1.00	4000
waw	1.00	1.00	1.00	4000
ya	1.00	1.00	1.00	4000
yaa	1.00	1.00	1.00	4000
za	1.00	1.00	1.00	4000
zay	1.00	1.00	1.00	4000
zero	1.00	1.00	1.00	4000
accuracy			1.00	176000
Macro avg	1.00	1.00	1.00	176000
Weighted avg	1.00	1.00	1.00	176000

Figure 9. Tabulation of precision, recall, f1-score, and support for each class for testing & training network of multi-model DenseNet121 & VGG16

Table 3 compares the validation accuracy and test accuracy of a single model and multi-model with epochs equal to 5 for the ASL dataset. The table shows the comparison between the performances of the single model and multi-model. It appears that the multi-model is better in feature extraction and classification than the single models. In addition, 100% accuracy was obtained in each of the training, validation and testing of the multi-model if the training was increased at epochs equal 7.

Table 4 shows the comparison between this work and previous works for the ASL dataset. From Table 4, it is clear that the proposed method, whether using a single model or a multi-model, is better than the models presented in the previous studies referred to in the table.

Table 3. Comparison of the validation accuracy and test accuracy for single and multi-models with Epochs=5 for ASL dataset

Model	Incorrect Recognize Training	Incorrect Recognize Validation	Validation Accuracy%	Incorrect Recognize Test	Test Accuracy%
DenseNet121	10	1	99.99	0	**100**
VGG16	38	6	99.93	3	99.97
DenseNet121&VGG16	2	0	**100**	0	**100**

Table 4. Comparison this work and previous works for ASL dataset

Authors	Description	Accuracy
Lum *et al*., 2020 [30]	Transfer learning using MobileNetV2 on 29 classes	98.67%
Sinha *et al*., 2019 [31]	Custom CNN model with fully connected layer on 29 classes	96.03%
Kadhim *et al*., 2020 [32]	Transfer learning using VGG1 on 28 classes	98.65%
Paul *et al*. 2020 [33]	Custom CNN model with fully connected layer on 24 classes	99.02%
Mahmud *et al*., 2018 [34]	HOG feature extraction & KNN classifier on 26 classes	94.23%
Prasad 2018 [35]	Image magnitude gradient for feature extraction on 24 classes	95.40%
Phong &Ribeiro 2019 [36]	Transfer learning on multiple architecture, etc on 29 classes	99.00%
Ashiquzzaman *et al*., 2020 [37]	Transfer learning using VGG16 on 29 classes	94.00%
This work single model	Transfer learning using DenseNet121on 29 classes	100.00%
This work single model	Transfer learning using VGG16 on 29 classes	99.97%
This work multi-model	Transfer learning using multi-model DenseNet12 & VGG16 on 29 classes	100.00%

5. CONCLUSION

Through analysis and discussion of the results of the proposed method, and under the limitations adopted by the research, the following was concluded: The research prepared about 220 thousand colour image datasets, as there is no public colour dataset for Arabic sign language recognition. When comparing the performance of single models and multi-models, it appears that most multi-models are better in feature extraction than single models. The DenseNet121 is the best CNN model for extracting features and classifying the Arabic sign language by depending on the total number of incorrectly recognized sign images in training, validation and testing datasets. Furthermore, based on the total number of incorrectly recognized sign images in training, validation, and testing datasets, the DenseNet121 & VGG16 multi-model CNN is the best for extracting features and classifying Arabic sign language. The multi-model is better for the feature extraction and classification of ASL than the single models by using the proposed method. And the accuracy of the proposed method, whether using a single model or a multi-model, is better than the models presented in the previous studies in extracting features and classifying ASL. In future researches, the work will be extended to develop a mobile-based application to recognize Arabic sign language in real-time. And also, the system will be extended to use dynamic gesture recognition for Arabic sign language, which requires preparing a video-based dataset.

REFERENCES

[1] A. Thongtawee, O. Pinsanoh, and Y. Kitjaidure, “A Novel Feature Extraction for American Sign Language Recognition Using Webcam,” *11th Biomedical Engineering International Conference (Bmeicon),* 2018, pp. 1-5, doi: 10.1109/BMEiCON.2018.8609933.

[2] A. Al-Khalifa, “The Arabic Dictionary of Gesture for the Deaf,” Supreme Counical for Family Affairs, 2008. [Online]. Available: https://arab.org/directory/supreme-council-family-affairs/

[3] M. Mukushev, A. Sabyrov, A. Imashev, K. Koishibay, V. Kimmelman, and A. Sandygulova, “Evaluation of Manual and Non-Manual Components for Sign Language Recognition,” *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 6073-6078.

[4] H. Cooper, B. Holt, and R. Bowden, “Sign Language Recognition,” *Visual Analysis of Humans*. London: Springer, pp. 539-562, 2011, doi: 10.1007/978-0-85729-997-0_27.

[5] A. 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.

[6] S. M. Elatawy, D. M. Hawa, A. A. Ewees, and A. M. Saad, “Recognition System for Alphabet Arabic Sign Language Using Neutrosophic and Fuzzy C-Means,” *Education and Information Technologies*, vol. 25, pp. 5601-5616, 2020, doi: 10.1007/s10639-020-10184-6.

[7] S. Hayani, M. Benaddy, O. El Meslouhi, and M. Kardouchi, “Arab Sign Language Recognition with Convolutional Neural Networks,” *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, 2019, pp. 1-4, doi: 10.1109/ICCSRE.2019.8807586.

[8] R. G. Crespo, M. Khari, E. Verdú, M. Khari, and A. K. Garg, “Gesture Recognition of RGB and RGB-D Static Images Using Convolutional Neural Networks,” *International Journal of Interactive Multimedia & Artificial Intelligence*, vol. 5, no. 7, pp. 23-27, 2019, doi: 10.9781/ijimai.2019.09.002.

[9] A. Dadashzadeh, A. T. Targhi, M. Tahmasbi, and M. Mirmehdi, “Hgr-Net: A Fusion Network for Hand Gesture Segmentation and Recognition,” *IET Computer Vision*, vol. 13, no. 8, pp. 700-707, 2019, doi: 10.1049/iet-cvi.2018.5796.

[10] A. I. Shahin and S. Almotairi, “Automated Arabic Sign Language Recognition System Based on Deep Transfer Learning,” *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 10, pp. 144-152, 2019.

[11] E. Elsayed and D. R. Fathy, “Sign Language Semantic Translation System Using Ontology and Deep Learning,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no.1, pp. 141-147, 2020, doi: 10.14569/IJACSA.2020.0110118.

[12] A. Althagafi, G. Althobaiti, T. Alsubait, T. Alqurashi, "ASLR: Arabic Sign Language Recognition Using Convolutional Neural Networks," *IJCSNS International Journal of Computer Science and Network Security,* vol. 20, no. 7, pp. 124-129, 2020.

[13] G. Latif, N. Mohammad, R. AlKhalaf, R. AlKhalaf, J. Alghazo, and M. Khan, "An Automatic Arabic Sign Language Recognition System Based on Deep CNN: An Assistive System for the Deaf and Hard of Hearing," *Int. Journal of Computing and Digital Systems,* vol. 9, no. 4, pp. 715-724, 2020, doi: 10.12785/ijcds/090418.

[14] Y. Saleh, and G. F. Issa, "Arabic Sign Language Recognition Through Deep Neural Networks Fine-Tuning," *International Journal of Online and Biomedical Engineering,* vol. 16, no. 5, pp. 71-83. 2020, doi: 10.3991/ijoe.v16i05.13087.

[15] K. Wangchuk, P. Riyamongkol, and R. Waranusast, "Real-Time Bhutanese Sign Language Digits Recognition System Using Convolutional Neural Network," *ICT Express,* vol. 7, no. 2, pp. 234-238, 2020, doi: 10.1016/j.icte.2020.08.002.

[16] R. Kadhim and M. Khamees, "A Real-Time American Sign Language Recognition System Using Convolutional Neural Network for Real Datasets," *Tem Journal*, vol. 9, no. 3, pp. 937-947, 2020, doi: 10.18421/TEM93-14.

[17] M. M. Kamruzzaman, "Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network," *Wireless Communications and Mobile Computing,* 2020, doi: 10.1155/2020/3685614.

[18] D. Tasmere, B. Ahmed, and S. R. Das, "Real Time Hand Gesture Recognition in Depth Image Using CNN," *International Journal of Computer Applications*, vol. 174, no. 16, pp. 28-32, 2021, doi: 10.5120/ijca2021921040.

[19] S. Gu, M. Pednekar, and R. Slater, "Improve Image Classification Using Data Augmentation and Neural Networks," *SMU Data Science Review,* vol. 2, no. 2, 2019.

[20] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700-4708, doi: 10.1109/CVPR.2017.243.

[21] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv Preprint arXiv: 1409, 1556*, 2014.

[22] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2018, pp. 8697-8710, doi: 10.1109/CVPR.2018.00907.

[23] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2017, pp. 1251-1258, doi: 10.1109/CVPR.2017.195.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "Mobilenetv2: Inverted Residuals and Linear Bottlenecks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

[25] M. Tan, And Q. Le, "Efficientnet: Rethinking Model Scaling for Convolutional Neural Networks," *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019, pp. 6105-6114.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp.770-778, doi: 10.1109/CVPR.2016.90.

[28] F. Zhang *et al*., "Mediapipe Hands: On-Device Real-Time Hand Tracking," *arXiv Preprint arXiv: 2006, 10214,* 2020.

[29] M. Grandini, E. Bagli, and G. Visan, "Metrics for Multi-Class Classification: An Overview," *arXiv Preprint arXiv: 2008, 05756*, 2020.

[30] K. Lum, Y. H. Goh, and Y. B. Lee, "American Sign Language Recognition Based on MobileNetV2," *Advances in Science, Technology and Engineering Systems Journal*, vol. 5, no. 6, pp. 481-488, 2020, doi: 10.25046/aj050657.

[31] S. Sinha, S. Singh, S. Rawat, and A. Chopra, "Real Time Prediction of American Sign Language Using Convolutional Neural Networks," *International Conference on Advances in Computing and Data Sciences*, 2019, pp. 22-31, doi: 10.1007/978-981-13-9939-8_3.

[32] R. Kadhim, and M. Khamees, "A Real-Time American Sign Language Recognition System using Convolutional Neural Network for Real Datasets," *TEM Journal*, vol. 9, no. 3, pp. 937-943, 2020, doi: 10.18421/TEM93-14.

[33] A. J. Paul, P. Mohan, and S. Sehgal, "Rethinking Generalization in American Sign Language Prediction for Edge Devices with Extremely Low Memory Footprint," *2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, 2020, pp. 147-152, doi: 10.1109/RAICS51191.2020.9332480.

[34] I. Mahmud, T. Tabassum, M. P. Uddin, E. Ali, A. M. Nitu, and M. I. Afjal, "Efficient Noise Reduction and HOG Feature Extraction for Sign Language Recognition,"*2018 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, 2018, pp. 1-4, doi: 10.1109/ICAEEE.2018.8642983.

[35] M. M. Prasad, "Gradient Feature based Static Sign Language Recognition," *International Journal of Computer Sciences and Engineering,* vol. 6, no.12, pp. 531-534, 2018. doi: 10.26438/ijcse/v6i12.531534.

[36] N. H. Phong and R. Bernardete, "Advanced Capsule Networks via Context Awareness," *International Conference on Artificial Neural Networks*, 2019, pp. 166-177, doi: 10.1007/978-3-030-30487-4_14.

[37] A. Ashiquzzaman, H. Lee, K. Kim, H. Y. Kim, J. Park, and J. Kim, "Compact Spatial Pyramid Pooling Deep Convolutional Neural Network Based Hand Gestures Decoder," *Applied Sciences,* vol. 10, no. 21, pp. 1-22, 2020, doi: 10.3390/app10217898.