International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 9, No.4 (July-2020)

http://dx.doi.org/10.12785/ijcds/090418

An Automatic Arabic Sign Language Recognition System based on Deep CNN: An Assistive System for the Deaf and Hard of Hearing

Ghazanfar Latif [1*], Nazeeruddin Mohammad [2], Roaa AlKhalaf [1], Rawan AlKhalaf [2], Jaafar Alghazo [3] and Majid A. Khan [1]

[1]Department of Computer Science, Prince Mohammad bin Fahd University, Saudi Arabia.
[2]Department of Information Technology, Prince Mohammad bin Fahd University, Saudi Arabia.
[3]Department of Computer Engineering, Prince Mohammad bin Fahd University, Saudi Arabia.

Received 12 Dec. 2019, Revised 12 Mar. 2020, Accepted 11 May 2020, Published 1 Jul. 2020

Abstract: People with disabilities have long been ignored. With the advancement of recent technologies, so many tools and software are designed for disabled people to improve their lives. In this research, the Arabic Sign Language (ArSL) recognition system is developed using the proposed architecture of the Deep Convolutional Neural Network (CNN). The aim is to help people with hearing problems to communicate with normal people. The proposed system recognizes the signs of the Arabic alphabet based on real-time user input. The Deep CNN architectures were trained and tested using a database of more than 50000 Arabic sign images collected from random participants of different age groups. Several experiments are performed with changing CNN architectural design parameters in order to get the best recognition rates. The experimental results show that the proposed Deep CNN architecture achieves an excellent accuracy of 97.6%, which is higher than the accuracy achieved by similar other studies.

Keywords: Arabic Sign Language, Assistive System, Convolutional Neural Networks (CNN), Deep Learning, Sign Language Recognition, Assistive Technology, Deaf, Hard of Hearing

1. INTRODUCTION

Persons with disabilities have different conditions, whether mental or physical, that result in various limitations in their senses or/and movements. As a result, persons with disabilities often have fewer educational opportunities and are less engaged in society than normal people. According to the United Nations (UN), people with disabilities represent fifteen percent of the world population [1]. Consequently, improving the quality of life for them is a major concern for governments, welfare organizations, and researchers. Particularly, the deaf and hard of hearing been given a lot of attention in the last decade. According to the World Health Organization (WHO), the number of people with disabling hearing loss is 360 million worldwide which is nearly five percent of the world population [2]. A lot of research has been conducted to understand the barriers faced by the deaf and hard of hearing that prevent them from engaging in their societies.

This research aims to design and implement an Arabic Sign Language (ArSL) recognition system that will ease the communication between deaf and normal people. Plenty of sign language recognition systems were implemented based on the American Sign Language (ASL) and the British Sign Language (BSL). However, ArSL did not get a lot of attention despite the widespread use of the Arabic language and the noticeable growth in the number of Arabic language speakers. The Arabic language is the native language of people in more than 20 countries across Africa and the Middle East with nearly 300 million native speakers worldwide according to the American Councils for International Education [3]. This geographical and demographical spread of the Arabic language underlines the fact that the insufficient attention given to Arabic Sign Language has to be reconsidered. So, this research mainly focuses on the recognition of ArSL alphabets. Sign language recognition systems are mainly concerned with finding a device, or a system, that plays the role of an interpreter in the process of communication between deaf and normal people [4]. These devices fall into the category of Human Computer Interaction (HCI) systems. Furthermore, any sign language recognition system can be implemented either

E-mail: glatif@pmu.edu.sa, nmohammad@pmu.edu.sa, 201401585@pmu.edu.sa , 201301954@pmu.edu.sa, jghazo@pmu.edu.sa, makhan@pmu.edu.sa

by using a sensor-based approach or an image-based approach as discussed in [5].

In sensor-based or glove-based systems, the user is required to wear a glove in order for the system to recognize hand gestures. The idea is based on interfacing multiple sensors with a glove to read the gesture to be then recognized by the system or the computer that is interfaced with the sensors. Sensor-based systems can be reliable and accurate; however, it has an overhead of wearing a glove that is loaded by cables, sensors and other materials. On the other hand, image-based systems overcome that burden so wearing any kind of gloves is not required providing more flexibility to the user than sensor-based systems. The idea of image-based systems is to use image processing techniques to recognize signs. In this research, the image-based approach is used. This research focuses on ArSL recognition system that can be used in real-time translation. The system can translate ArSL signs into text and speech using deep CNN.

The rest of the paper is organized as follows. First, the introduction and background information is provided in section 2. Requirement analysis and system design are provided in sections 3 and 4. Finally, conclusions and future work are discussed in section 5.

2. LITERATURE REVIEW

Image-based recognition systems, in general, can be implemented using two approaches which are deep learning and feature extraction. In feature extraction based implementation, the features are hand-engineered from the images using feature descriptor to be fed to a learning algorithm for classification [6-10]. Consequently, these algorithms' performance is highly dependent on hand-engineered features. So if the given computer vision problem involves complex features, the progress in solving it will be obstructed due to the difficulty in extracting complex features [11]. On the other hand, in deep learning approaches, the features are extracted in an automated manner by applying a series of transformations on the input image [6]. First, in deep learning architectures the learning of features is hierarchical, so the extracted features are the most robust features in the input image which mean that complex problems are better modeled when deep learning architectures are used [12-15]. Second, researches have shown that state-of-the-art results are achieved when deep learning architectures with multiple layers are used and that it often outperforms feature-based computer vision architectures [16-19]. Convolutional Neural Networks (CNN) are based on deep learning architecture and are the most widely used approach to solve different problems such as image recognition [20]. CNNs are inspired by the visual system of humans in which various levels of processing are done in sequence; so in CNNs, multiple layers are stacked and designed to extract abstract features in a simulation of the process from the retina to the cortex [21]. This simulation gives CNNs the potential of performing well in image classification tasks which makes them a suitable choice for this research.

CNNs consist of convolutional layers alternating with subsampling layers, pooling layers, and followed by a fully connected layer; a brief explanation of CNN architecture will be discussed here based on [16-22]. In the convolutional layers, multiple kernels are utilized to convolve the input image to generate multiple feature maps. In pooling layers, the dimensions of network parameters and feature maps are reduced and these layers are translation invariant. Finally, the fully connected layers can be considered as the classifier portion of the network. The training of CNN consists of two stages which are the forward and the backward stages. In the forward stage, the input image is represented by the current parameters in each layer of the network. Once the output is predicted, the backward stage starts which is based on the Backpropagation algorithm. Backpropagation is a supervised learning algorithm that finds the minimum value of error function to update the weights. These weights are initialized randomly, then the error function is calculated. As the error rate which is the difference between the expected result and the output result is high, the weights are decreased. Then the weights will be decreased further until it becomes a minimum. This minimum value is found using the gradient descent algorithm, which starts with an initial value then iteratively moves toward the minimum.

Deep learning shows its potential in various fields; so it was implemented in various applications such as highways autonomous driving. Huval, B. et al. [23] used deep learning CNNs for detecting lane and vehicle in highways for autonomous driving. The authors believe that using traditional computer vision approaches, despite their potential in many applications, on highway autonomous driving will not produce the desired performance.

Besides using deep learning architecture in autonomous driving, it was used in extracting the desired features in many applications. Chen et al. presented a 3-D CNN used for deep Feature Extraction (FE) for Hyper-Spectral Image (HSI) classification [21]. Their regularized deep FE method that is based on 3-D CNN aimed to extract spectral and spatial features of HSIs. They used three of the most commonly used data sets for HSIs which are Indian Pines, University of Pavia, and Kennedy Space Center. The authors discussed the overfitting problem, and explained how they resolved it by using the Rectified Linear Unit (ReLU) and dropout. For further improvements, the authors used virtual samples, that are generated from the actual training samples, to train the CNN in addition to the training samples. Finally, the authors believe that 3-D CNN performed exceptionally well in HSI FE although the number of training samples was limited.

Moreover, deep learning also shows its potential in recognition tasks not only when the inputs are given as images, but also for various input shape such as speech signals. Researchers in [24] overviewed the work done on speech recognition using deep learning approaches since 2009. The replacement of Gaussian Mixture Model (GMM) based Hidden Markov Model (HMM) systems with deep learning approaches such as Deep Belief Nets (DBNs), Deep Neural Nets (DNNs) and Deep AutoEncoders (DAEs) showed a massive potential since the de-correlation of feature components becomes irrelevant. The speech recognition error rate had decreased noticeably when the large scale DNNs were used as a replacement for Mel-Frequency Cepstral Coefficients (MFCC). Additionally, the change from using MFCC to spectral features allowed the use of deep CNNs for the TIMIT set in the phone recognition experiment and the error rate dropped from 20.4% to 19.7%. The error rate dropped further after regularizing the CNN with dropout approach to 18.7%. The authors believe that DNNs can learn features effectively and is capable of handling heterogeneous data from multiple languages and acoustic sources.

Furthermore, deep learning was used for the task of object detection and it showed promising results in this area. Shuo Yang et al. [25] presented their Deep Convolutional Network (DCN) for face detection. The image is given as an input to five CNNs, deep layers are shared between them to save computational time. A partness map will be outputted by each CNN which indicates where a specific facial component is located in the image. By using object proposal methods, candidate windows are generated. These windows are ranked accordingly, using the extracted partness maps, to their faceness scores. By training a multi-task CNN, the candidate windows are refined and then the optimization of face classification and bounding box regression is done. The obtained results suggested that the system is robust enough to handle unconstrained images while training; and despite the use of uncrossed images, the generated response maps could precisely locate the face parts and the system is efficient in proposing candidate windows.

Another addition to deep learning in sign language was presented by Huang et al. [26]. Dataset consists of 26x3 recorded video clips from three different signers. Then, the frames were extracted from recorded video clips to a total of 65000 frame images. The training set has 52000 images and the testing set has 13000. Each sample image is converted to a feature vector of 66 dimensions. Those features are applied for training DNN directly. Three DBN hidden layers were built with each having 500, 500, and 2000 hidden units. The input units are 66, and the output units are 26, corresponding to the 26 alphabet signs. The average accuracy rate of recognizing 26 alphabet signs is 98.9%. However, sign recognition systems based on mouth shapes are also present. For instance, Koller and Bowden [27] proposed a sign language recognition system in terms of mouth shapes. The proposed system incorporates CNN classifier outputs into the HMM approach which allows iterative learning of CNNs and forced temporal alignment on video.

Despite the high accuracies obtained and the outstanding designs, the authors believe that more focus should be given to how to use these systems in real-time situations. Additionally, these systems should be capable of translating the recognized signs to different formats, such as text and speech, to be able to accommodate all needs.

3. PROPOSED METHODOLOGY

The overall concept of this research is to design a deep CNN that recognizes the ArSL alphabets with high accuracy. Figure 1 shows a high-level view of the system. The collected data set is given as an input to the proposed CNN architecture for training, validation, and testing. CNNs consist of multiple hidden layers between the input and output layers. The proposed CNN is implemented using Python libraries. Once the CNN is trained, it is ready to be used for classifying new data which were not part of the collected data set. The user can perform the sign by her/his right hand in front of a camera, which will be forwarded to a computing system. The system will use the trained CNN, and outputs the input alphabet sign visually on screen and orally through the speakers.

The ArSL system depends on a group of standard Arabic alphabet signs that were approved by the fifth Arab Conference in Deaf Care in 1986. Some countries published their own sign language dictionary consisting of dependent words and phrases. The alphabet signs that are used in this paper are considered as a standard to all users of Arabic Sign Language despite their own local sign language [28]. They often achieve high accuracy rates in image recognition systems with a low error rate.

A. Arabic Sign Language (ArSL) Dataset

A new dataset consisting of 54,000 images of the ArSL alphabets performed by more than 40 people for 32 standard Arabic signs and alphabets and detailed in [29] was used in this study. The images were RGB images with different dimensions which led to the need for pre-processing the images before feeding them to the network. The data set was randomly partitioned into eighty percent training set, and twenty percent testing set. The training set was further partitioned with twenty percent of the training images used for validation. The total number of output classes is 32 that ranges from 0 to 31 each represents an ArSL sign as samples shown in Figure 3. Table 1 shows the different classes with their corresponding labels, names and number of images.

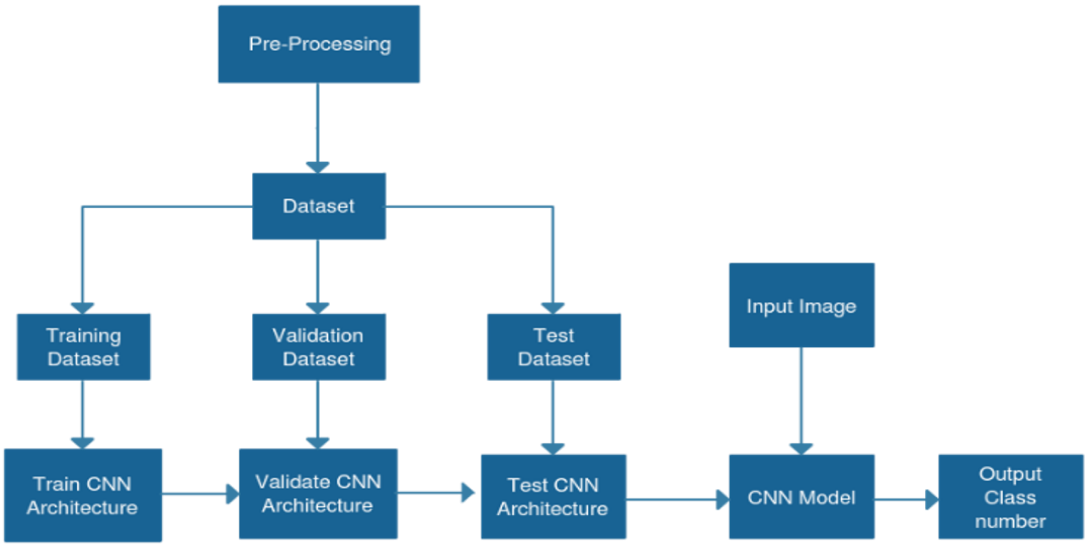

Figure 1. High-level architecture of the proposed system

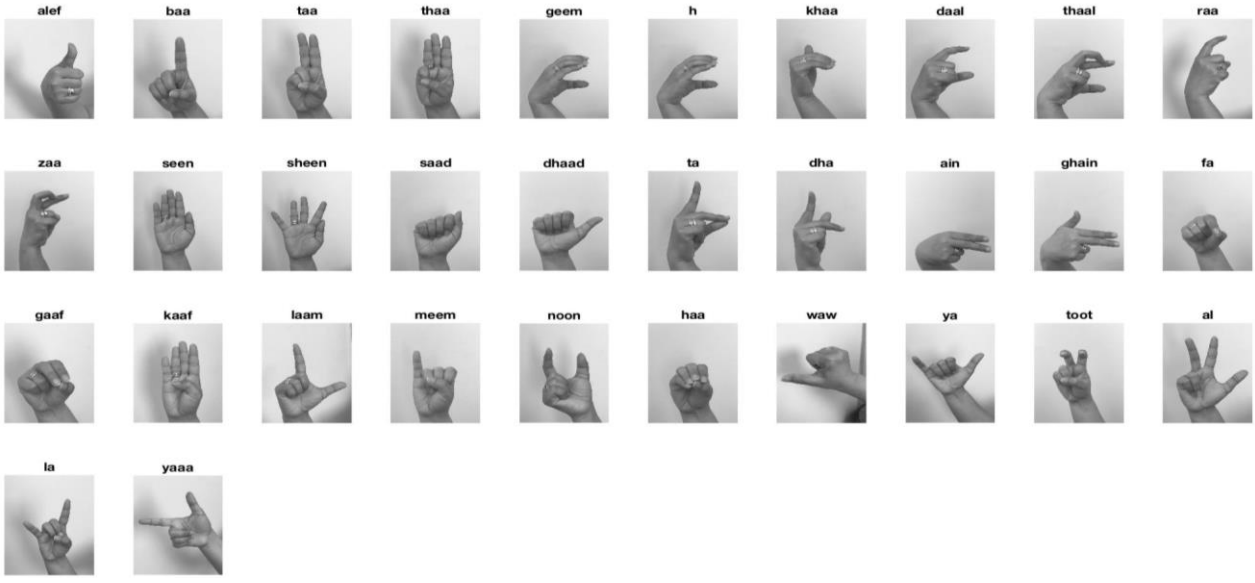

Figure 2. Representation of the Arabic Sign Language for Arabic Alphabets

B. Preprocessing

Before feeding the dataset to the network, all sign images are transformed to grayscale 64x64 dimension to get rid of the extra processing needed for 3 color channels in RGB images. For the efficiency and speed of the training, the format of the images was converted from the int8 format to float32. All images were normalized so that the pixel values range from 0 to 1. The dataset was reshaped to meet the format requirements so that it can be fed to the CNN.

Additionally, since CNN cannot deal directly with categorical input, the class labels were converted into one-hot encoding vectors using the One-hot encoding method [30]. By doing so, a vector is generated which has one row and as many columns as the number of output classes, 32 columns, so for each image it has a dimension of 1x32. For each image in the data set, this vector has the value zero in all indexes except the index that corresponds to the image's class, it has a value of 1.

As mentioned above, the data set was partitioned into training, validation and testing sets. This partitioning was done for cross-validation which is a statistical approach to

evaluate the performance of Machine Learning (ML) algorithms. The Hold-out cross-validation technique was used which basically splits the data set into three mutually independent sets. This technique is commonly used due to its efficiency and ease of implementation [31]. Since there is no fixed rule on the percentage in which the data set is partitioned by, one of the commonly used partitioning percentages was chosen in this study, which is and 20% for testing and 80% for the training set of which 20% is taken for validation,.

TABLE I. SUMMARY OF EXTRACTED ECG CLASSES DATA

#	Class	# of Images	#	Class	# of Images
0	أ(alef)	1672	16	ظ(dha)	1723
1	ب (ba)	1791	17	ع(ain)	2114
2	ت (taa)	1838	18	غ(ghain)	1977
3	ث (thaa)	1766	19	ف(fa)	1955
4	ج (jeem)	1552	20	ق (gaaf)	1705
5	ح (haa)	1526	21	ك(kaaf)	1774
6	خ(kha)	1607	22	ل(laam)	1832
7	د (daal)	1634	23	م(meem)	1765
8	ذ (thaal)	1582	24	ن(noon)	1819
9	ر (raa)	1659	25	ه(ha)	1592
10	ز (zaa)	1374	26	و(waw)	1371
11	س (seen)	1638	27	ي(yaa)	1722
12	ش (sheen)	1507	28	ة(taaa)	1791
13	ص(saad)	1895	29	ال(al)	1343
14	ض(dhaad)	1670	30	لا(la)	1746
15	ط(ta)	1816	31	ئ (yaaa)	1293

C. Proposed CNN Architecture

The design of the proposed CNN consists mainly of two stages. The purpose of the first stage was to obtain preliminary results that will guide us in the next experiments based on Figure 2. Based on initial results, multiple experiments are done in the second stage by changing different design parameters in order to come up with the most suitable design for the proposed ArSL system.

The main building blocks of any CNN are the convolutional and pooling layers. Therefore, as initial design, one of the most commonly used designs was initially chosen, which is to have three convolutional and pooling layers. The obtained accuracy using this initial architecture will help in the design process. Figure 2 shows the initial CNN architecture consists of three convolutional layers, three pooling layers, and a fully connected layer. The details of each layer are as follows.

- The first convolution layer: The convolution operation is done using a kernel of size 3x3. Since this layer has 32 kernels, there will be 32 generated feature maps.
- The second convolution layer: The convolution operation is done using the same kernel size as in the first layer; however, it has 64 kernels. So there will be 64 generated feature maps.
- The third convolution layer: There are 128 kernels of the same size used in the previous layers to apply the convolution operation. So, 128 feature maps are generated.
- The Pooling Layers: There is one pooling layer after each convolution layer, each has a window size of 2x2.

The proper choice of different design hyper-parameters, such as non-linearity and pooling variants, directly affects the performance of the network. Since there is no clear guidance on how to choose the CNN hyper-parameters, many researchers tend to use trial and error experimentation in order to discover the best settings [32]. Previous research studies that have been carried in the area of deep learning generally and sign language recognition particularly were analyzed in order to choose the hyper-parameters setting that is more likely suitable for this research. By analyzing the work done in [33-35], the hyper-parameters setting is as follows.

- Pooling Layer: It helps in reducing the number of computations needed in the training process by reducing the dimension of the image, and therefore the overfitting problem is reduced. The max-pooling technique was used since the convergence rate is faster compared to other subsampling techniques, and thus has a better generalization performance. The maximum pixel value in a non-overlapping region (which is equal to the window of the pooling) is the output of this layer; this is beneficial in creating position invariance.
- Non-Linearity: Since the relation between the images and their classes is not linear, introducing non-linearity in the CNN is needed. This is achieved by using non-linear activation function in order to build non-linear relation between the images and their classes are through CNN. The Rectified Linear Unit (ReLU) is a widely used activation function in CNN. ReLU has the advantage that the CNN trains faster compared to other functions. One of the ReLU variants was used which is the leaky ReLU since it overcomes the problem of dead neurons, which is a common issue with the original ReLU. Basically, leaky ReLU does not output zero when the input values are less than zero, instead, it outputs a negative value. So after each convolution layer, and before the pooling layer, a leaky ReLU activation function was added.
- Dense Layer Activation Function and Loss Function: As mentioned above, The leaky ReLU activation function was used multiple times in the proposed network. However, the output of the fully connected

layer, which is a one-dimensional vector, is passed to a softmax activation function to predict the label of that particular input. The input vector is transformed into a vector of the same size but the values range from zero to one only and the summation is always equal to 1. Softmax function outputs a probability distribution that is then converted to a one-hot encoding vector. Therefore, the element that has the largest portion of probability distribution will have the value one in the one-hot vector and all other elements will have zero. To estimate the loss of softmax, the categorical cross-entropy function is used.

After setting the needed hyper-parameters, the network was compiled using Adam optimizer. The need of using optimization algorithms is the nature that CNN weights and biases are set automatically, and they are of great importance in the field of machine learning. They work on optimizing a given function; whether maximizing or minimizing it with respect to its parameters. Since the loss function in CNN is differentiable with respect to its parameters, gradient descent-based algorithms are often used; first order partial derivatives are also fast to compute. Adam is an optimization algorithm that is used to update network weights. It is computationally efficient, requires less memory and has straightforward implementations. Given different parameters, this algorithm computes each parameter's adaptive learning rate by estimating the gradient's first and second moments [36]. Table 2 shows a summary of the parameters in each layer as well as the total parameters in the proposed network. The CNN input layer consists of 3 channels RGB with dimensions of 64x64. The first convolutional layer has 32 feature maps having a kernel size of 5x5. The other two convolution layers have the same design with different feature map sizes. After each convolutional layer, the leaky ReLU activation function is applied to convert negative to near zero values in order to speed up the training. In the same manner, the 2x2 max pooling layer is applied to reduce the features by half. Flatten layer is used to convert multidimensional feature maps to a one-dimensional feature map. Finally, the dense layer is used to map to the output results.

TABLE 2. SUMMARY OF THE INITIAL PROPOSED CNN NETWORK

Network Layer	Output Shape	Parameters
Convolution 1	(64, 64, 32)	320
Leaky ReLU Layer 1	(64, 64, 32)	0
Max Pooling 1	(32, 32, 32)	0
Convolution 2	(32, 32, 64)	18496
Leaky ReLU Layer 2	(32, 32, 64)	0
Max Pooling 2	(16, 16, 64)	0
Convolution 3	(16, 16, 128)	73856
Leaky ReLU Layer 3	(16, 16, 128)	0
Max Pooling 3	(8, 8, 128)	0
Flatten Layer	(8192)	0
Dense Layer 1	(128)	1048704
Leaky ReLU Layer 4	(128)	0
Dense Layer 1	(32)	4128
Total parameters: 1,145,504		

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

The initial architecture of CNN trained was trained for 50 epochs with a batch size of 64. Figure 4 plots the variations in the training accuracy, training loss, validation accuracy and validation loss with respect to the epoch number. The test accuracy of the initial architecture was 80.31%, which does not meet the accuracy requirement, and it is not reliable for real-time recognition. Consequently, different design parameters were changed to test the effect of these changes on test accuracy. A total of 15 different experiments were performed on the CNN architecture in order to finalize the most suitable design for this research. Note that all the preprocessing steps remain the same as described above.

The results are shown in Table 4. It is clear that the best design for the proposed system is to use four convolutional layers with fixed kernels size of 3x3, four maxpooling layers with window size 2x2 and five dropout layers. Table 3 shows the improved CNN network summary which produced better results.

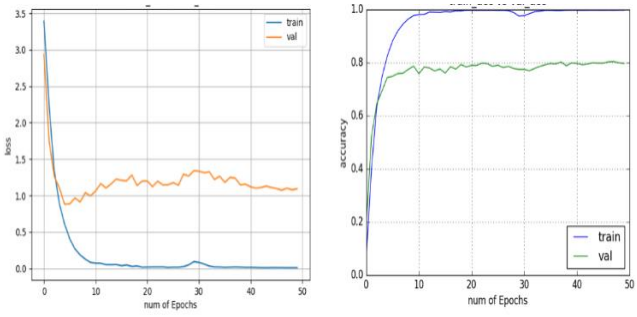

Figure 3. (A). Training Loss vs. Validation Loss, (B). Training Accuracy vs. Validation Accuracy

First, some experiments were performed to show the effect of increasing the number of images in the data set on the test accuracy. By increasing the images from 8302 to 27985, the test accuracy increased from 80.3% to 93.9%. Further increase in the number of images to 33406 and 50,000 led to an increase in the accuracy to 94.1% and 95.9% respectively. The changes in the accuracy are obtained by only changing the number of images in the data set while keeping all other design parameters constant. It can be clearly seen that increasing the size of the data set is positively affecting the test accuracy. However, the percentage of increase is small as more and more images are added. As clearly shown in Figure 5, increasing the images from 8,302 to 27,985 increased the accuracy by 0.136% only with an accuracy rate of 93.9%. The full dataset 54,049 increased the accuracy to 95.9%.

Thus, the number of images beyond a certain point starts to have little effect on accuracy using the proposed method in this paper. Further, adding more images to the data set is a challenge since the number of participants who are willing to perform the signs is limited. In the developed prototype, the maximum number of images obtained in the data set is 50,000; however, for the upcoming prototypes, further increase of the number of images will be done by applying different methods such as data augmentation. In addition to the limitation of dataset size and the limitation of the number of participants used in obtaining the dataset, there are other limitations in this particular method. There was a limitation in the system hardware available as image processing and deep learning algorithms require high processing and memory requirements. The limitation of producing reasonable recognition within reasonable time and accuracy was also a limitation.

TABLE 3. SUMMARY OF THE IMPROVED CNN NETWORK

Network Layer	Output Shape	Parameters
Convolution 1	(64, 64, 32)	320
Leaky ReLU Layer 1	(64, 64, 32)	0
Max Pooling 1	(32, 32, 32)	0
dropout_1 (Dropout)	(32, 32, 32)	0
Convolution 2	(32, 32, 64)	18496
Leaky ReLU Layer 2	(32, 32, 64)	0
Max Pooling 2	(16, 16, 64)	0
dropout_2 (Dropout)	(16, 16, 64)	0
Convolution 3	(16, 16, 128)	73856
Leaky ReLU Layer 3	(16, 16, 128)	0
Max Pooling 3	(8, 8, 128)	0
dropout_3 (Dropout)	(8, 8, 128)	0
Convolution 4	(8, 8, 256)	295168
Leaky ReLU Layer 4	(8, 8, 256)	0
Max Pooling 4	(4, 4, 256)	0
dropout_4 (Dropout)	(4, 4, 256)	0
flatten_1 (Flatten)	(4096)	0
dense_1 (Dense)	(128)	524416
Leaky ReLU Layer 5	(128)	0
dropout_5 (Dropout)	(128)	0
dense_2 (Dense)	(32)	4128
Total parameters: 916,384		

Second, comparing the training and validation accuracy of the initial architecture, it can clearly be seen that the model is overfitting. Overfitting is a situation in which the CNN model fits perfectly on the training data. This results in high training accuracy but too low test and validation accuracies. This can be handled using many approaches; a convolutional layer was used in the proposed architecture. Different kernel sizes were used but the result did not show improvement in the test accuracy. The number of epochs were reduced to 30 and dropout layers added after each maxpooling layer. Dropout is a regularization method that is used to reduce the overfitting problem. It randomly drops out units as well as their connections during training by preventing co-adaptions between training data. For this study, the range of its parameter were restricted between 0.2 and 0.5 in instead of 0 and 1.

Third, the behavior of the convolutional layers is determined by the number of layers and the size of kernels in each layer. To begin with, the size of the kernels used is dependent on what assumption was made on the input images. In this research, the nature of the input images is highly structural since they all represent a sign performed by the right hand. The best way to differentiate the different signs is by focusing on small and local features by convolving with small kernels. Using kernels of size 1x1 means that the convolution operation is done pixel-wise; so it is not practical since neighboring pixels are affected by each other. Small size kernels are believed to be most suitable for the proposed design, despite that, experiments on the effect of using lager kernels were performed. In the data set that contains 27985 images, experiments with 3x3, 5x5 and 7x7 kernels were performed. All other design parameters are kept the same as in the initial design. The obtained results are 94.1%, 94.4%, and 94.1% respectively.

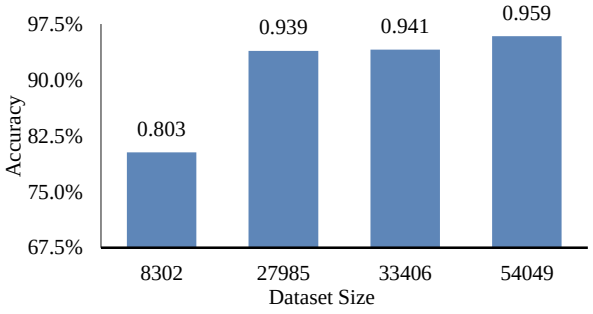

Figure 4. The effect of increasing the number of images on the accuracy

The observation is that the kernel size 7x7 is not suitable since it led to a decrease in the test accuracy compared to 5x5 kernels. It also gave the same accuracy as when 3x3 kernels were used but it took more training time. To validate this, the same three tests were performed with the same kernel size but on an updated data set with 33406 images. The obtained results are 95.6%, 94.7% and 94.1% for each kernel size respectively. So it can be seen clearly that using 7x7 kernels is not reflecting any improvement in the test accuracy. Hence, 7x7 kernels were not applied in the designs; with only 3x3 and 5x5 kernels to be considered. A related point to consider is to use different kernel sizes in each convolutional layer. An experiment using different kernel sizes was performed

but the result did not show improvement in the test accuracy.

Fourth, the number of hidden layers in any CNN contributes to the test accuracy. As it is generally known that the deep architecture of CNN is the core reason behind its strength. In order to know the maximum number of layers that can be stacked in the proposed CNN that will give the maximum accuracy, several tests were performed and the results are shown in Table 4.

TABLE 4. SUMMARY OF RESULTS FOR DIFFERENT CNN LAYERS WITH DIFFERENT PARAMETERS

Test Acc.	Data Set	Conv. Layers				Pooling Layers	Dropout Layers				
95.3%	27985	Conv1 32x(3x3)	Conv2 64x(3x3)	Conv3 128x(3x3)	Conv4 256x(3x3)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)
94.3%	27985	Conv1 32x(5x5)	Conv2 64x5x5)	Conv3 128x(5x5)	Conv4 256x(5x5)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)
95.8%	33406	Conv1 32x(3x3)	Conv2 64x(3x3)	Conv3 128x(3x3)	Conv4 256x(3x3)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)
95.2%	33406	Conv1 32x(5x5)	Conv2 64x5x5)	Conv3 128x(5x5)	Conv4 256x(5x5)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)
97.6%	50,000	Conv1 32x(3x3)	Conv2 64x(3x3)	Conv3 128x(3x3)	Conv4 256x(3x3)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)
97.1%	50,000	Conv1 32x(5x5)	Conv2 64x5x5)	Conv3 128x(5x5)	Conv4 256x(5x5)	4x(2x2)	Dropout1 (0.25)	Dropout2 (0.25)	Dropout3 (0.4)	Dropout4 (0.25)	Dropout5 (0.3)

It can be observed from Table 4 that increasing the number of hidden layers improved the test accuracy when using 3x3 kernels. Therefore, this design will be tested on the updated data set to verify this observation. Furthermore, the effect of adding an extra layer for the 33406 data set was tested while using 3x3 kernels. The result of this change is a decrease in the test accuracy as compared to the previous highest accuracy in the same data set. Based on the results, we conclude that having four convolutional layers is the most suitable design.

Finally, 3 test were executed using the data set of 50,000 images. The first test was using the same CNN design as the initial design; this was intended to show the effect of adding more images on the test accuracy. Second, based on the previous analysis on previous designs on the data sets of 27985 and 33406 images, two tests are performed to determine the best design. The last two rows of Table 4 show a summary of these tests. From these results, it is clear that the best design for the proposed ArSL system is to use four convolutional layers with fixes kernels size of 3x3, four maxpooling layers with window size 2x2 and five dropout layers. Computational training time of each experiment was also calculated which is summarized in Table 5.

TABLE 5. COMPARISON OF COMPUTATION TIME FOR DIFFERENT DATASET SIZES WITH THEIR ACCURACIES

Accuracy	Dataset	Epochs	Kernel Size	Training Time
80.3%	8302	50	3x3	55.2 minutes
94.3%	27985	50	3x3	2.6 hours
95.2%	33406	50	3x3	3.8 hours
97.1%	50000	50	3x3	6.2 hours
97.6%	50000	100	3x3	8.1 hours

Table 5 shows a comparison of the proposed method in this paper compared with several methods from recent literature. It is clear from the table that the proposed method outperforms those in previous literature on similar datasets but not the same dataset used in this paper. This paper uses the dataset published in [29] which is relatively a new dataset and has not been explored by any other research because it is very recent. Thus, exact comparison is not possible.

5. CONCLUSION

Arabic Sign Language (ArSL) recognition system was designed using Convolutional Neural Networks (CNN). The training and testing dataset consist of more than 50000 sign samples collected from more than 40 random participants. The dataset passed through the preprocessing stage before feeding it to the network. It got partitioned into training, testing and validating datasets. implemented total of 15 CNN experiments were implemented with changing different design parameters. The initial design was based on similar previous work which was used as a base for the subsequent designs. Comparing the test accuracy of the same design implemented on different data sets led to the best design in terms of accuracy. It consists of four convolutional layers, four maxpooling layers, and five dropout layers. A 97.6% accuracy was obtained. The usefulness of the designed CNN architecture was demonstrated by implementing a practical system that takes real-time ArSL signs performed by a user via webcam, classifies them to the corresponding letters, and then gives a visual and oral output. The system will be useful for usage by hearing loss or deaf people. Future work will include an enhanced system to translate the words and sentences.

Future work would include designing algorithms that would produce more accuracy as well as increasing the dataset size which would increase the variety of images in terms of lighting, noise, etc. The ultimate goal of this research is to have a working prototype of a handheld device that could automatically translate Arabic sign language in real-time which would enhance the communication between persons who are deaf and hard of hearing problems with their environment.

REFERENCES

[1] Skempes, Dimitrios, Gerold Stucki, and Jerome Bickenbach. "Health-related rehabilitation and human rights: analyzing states' obligations under the United Nations Convention on the Rights of Persons with Disabilities." *Archives of physical medicine and rehabilitation*, Volume 96,2015, pp. 163-173.

[2] Krak, Iu, et al. "Information technology for automated translation from inflected languages to sign language." *Recent Advances in Information Technology*. CRC Press, 2017, pp. 51-82.

[3] Al-Musnad, Bayan Ibrahim. "The Role of Motivation and Attitude in Second Language Learning: A study of Arabic Language Learning among Foreign Female Nurses in Riyadh, Saudi Arabia." *Journal of Applied Linguistics and Language Research*, Volume 5, 2018, pp. 157-183.

[4] Mohandes, Mohamed, Junzhao Liu, and Mohamed Deriche. "A survey of image-based arabic sign language recognition." *2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSD14)*. IEEE, 2014.

[5] Assaleh, Khaled, and M. Al-Rousan. "Recognition of Arabic sign language alphabet using polynomial classifiers." *EURASIP Journal on Advances in Signal Processing*, Volume 13, 2005, 507614.

[6] Lee, Andy. "Comparing deep neural networks and traditional vision algorithms in mobile robotics." *Swarthmore University* (2015).

[7] Er-Rady, Adil, et al. "Automatic sign language recognition: A survey." *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. IEEE, 2017.

[8] Alghazo, Jaafar M., et al. "Multi-Language Handwritten Digits Recognition based on Novel Structural Features." *Journal of Imaging Science and Technology*, Volume 63, 2019, pp. 20502-1.

[9] Latif, Ghazanfar, DNF Awang Iskandar, Jaafar M. Alghazo, and Nazeeruddin Mohammad. "Enhanced MR image classification using hybrid statistical and wavelets features." *IEEE Access*, Volume 7, 2018, pp. 9634-9644.

[10] Alghazo, Jaafar M., et al. "An Online Numeral Recognition System Using Improved Structural Features–A Unified Method for Handwritten Arabic and Persian Numerals." *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, Volume 9, 2017, pp. 33-40.

[11] Srinivas, Suraj, et al. "A taxonomy of deep convolutional neural nets for computer vision." *Frontiers in Robotics and AI*, Volume 2, 2016, pp. 1-36.

[12] Oyedotun, Oyebade K., and Adnan Khashman. "Deep learning in vision-based static hand gesture recognition." *Neural Computing and Applications*, Volume28, 2017, pp. 3941-3951.

[13] Zheng, Lihong, Bin Liang, and Ailian Jiang. "Recent advances of deep learning for sign language recognition." *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, 2017.

[14] Kalbhor, Snehal R., and Ashwini M. Deshpande. "Digit Recognition Using Machine Learning and Convolutional Neural Network." *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2018.

[15] Hossen, M. A., et al. "Bengali Sign Language Recognition Using Deep Convolutional Neural Network." *2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*. IEEE, 2018.

[16] Pandey, Gaurav, and Ambedkar Dukkipati. "To go deep or wide in learning?." *arXiv preprint arXiv:1402.5634* (2014).

[17] Latif, Ghazanfar, et al. "Deep convolutional neural network for recognition of unified multi-language handwritten numerals." *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*. IEEE, 2018.

[18] Vannocci, Marco, et al. "Flatness Defect Detection and Classification in Hot Rolled Steel Strips Using Convolutional Neural Networks." *International Work-Conference on Artificial Neural Networks*. Springer, Cham, 2019.

[19] Alghmgham, Danyah A., et al. "Autonomous Traffic Sign (ATSR) Detection and Recognition using Deep CNN." *Procedia Computer Science*, Volume 163, 2019, pp. 266-274.

[20] Hijazi, Samer, Rishi Kumar, and Chris Rowen. "Using convolutional neural networks for image recognition." *Cadence Design Systems Inc.: San Jose, CA, USA*, 2015, pp. 1-12.

[21] Chen, Yushi, et al. "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks." *IEEE Transactions on Geoscience and Remote Sensing*, Volume 54, 2016, pp. 6232-6251.

[22] Guo, Yanming, et al. "Deep learning for visual understanding: A review." *Neurocomputing*, Volume 187, 2016, pp. 27-48.

[23] Huval, Brody, et al. "An empirical evaluation of deep learning on highway driving." *arXiv preprint arXiv:1504.01716* (2015).

[24] Deng, Li, et al. "Recent advances in deep learning for speech research at Microsoft." *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013.

[25] Yang, Shuo, et al. "From facial parts responses to face detection: A deep learning approach." *Proceedings of the IEEE International Conference on Computer Vision*. 2015.

[26] Huang, Jie, et al. "Sign language recognition using real-sense." *2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*. IEEE, 2015.

[27] Koller, Oscar, Hermann Ney, and Richard Bowden. "Deep learning of mouth shapes for sign language." *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2015.

[28] Esposito, John L., and Emad El-Din Shahin, eds. *The Oxford handbook of Islam and politics*. Oxford University Press, 2016.

[29] Latif, Ghazanfar, et al. "ArASL: Arabic Alphabets Sign Language Dataset." *Data in brief*, Volume 23, 2019, Article 103777.

[30] Dubey, Abhishek, and Jules White. "Dxnat-deep neural networks for explaining non-recurring traffic congestion." *arXiv preprint arXiv:1802.00002* (2018).

[31] Reitermanova, Z. "Data splitting." *WDS*., Volume 10, 2010.

[32] Mishkin, Dmytro, Nikolay Sergievskiy, and Jiri Matas. "Systematic evaluation of convolution neural network advances on the imagenet." *Computer Vision and Image Understanding*, Volume 161, 2017, pp. 11-19.

[33] Nagi, Jawad, et al. "Max-pooling convolutional neural networks for vision-based hand gesture recognition." *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. IEEE, 2011.

[34] Horiguchi, Shota, Daiki Ikami, and Kiyoharu Aizawa. "Significance of softmax-based features over metric learning-based features." (2016).

[35] West, Nathan E., and Tim O'Shea. "Deep architectures for modulation recognition." *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*. IEEE, 2017.

[36] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

Ghazanfar Latif is Research Coordinator, at Prince Mohammad bin Fahd University, Deanship of Graduate Studies and Research. He holds PhD degree from University of Malaysia Sarawak, Malaysia. He earned his MS degree in Computer Science from King Fahd University of Petroleum and Minerals, Saudi Arabia in 2014 and BS degree in Computer Science from FAST National University of Computer and Emerging Sciences, Pakistan in 2010 by remaining in Dean's honor list. Throughout his educational carrier, he got a number of achievements like full scholarship for FSc, BS-CS and MS-CS. His research interests include Image Processing, Artificial Intelligence, Neural Networks, and Medical Image Processing.

Nazeeruddin Mohammad completed his PhD from the School of Computing and Information Engineering at the University of Ulster, Coleraine, UK in 2007. He received his Bachelor of Engineering (B.E) degree in Electronics and Communications Engineering from Osmania University, India in 1996. He has an M.S. degree in Systems Engineering from King Fahd University of Petroleum & Minerals (KFUPM), KSA in 1999. He also received practical training and an Honours Diploma in Software Development from BDPS, India in 1997. Prior to the current position, he worked as an IT Infrastructure Manager/Administrator and Lecturer at KFUPM (KSA), as a postgraduate researcher at University of Ulster (UK), as a research intern at Cisco Systems (USA), and as an IT Infrastructure Manager/Administrator and Researcher at Portland House Research Group (Australia). He is a recipient of M.S. Research Scholarship from KFUPM (1997), Vice Chancellors Research Scholarship from University of Ulster (2004) and first prize in the Faculty of Engineering Business Plan Competition at the University of Ulster (2006).

Roaa AlKhalaf, Rawan AlKhalaf were students of Computer Engineer Department at College of Computer Engineering and Sciences, Prince Mohammad bin Fahd University, Saudi Arabia.

Jaafar Alghazo obtained his PhD and MSc in Computer Engineering from Southern Illinois University Carbondale in 2004 and 2000 respectively. He joined Prince Mohammad Bin Fahd University (PMU) as founding Dean of the College of Computer Engineering and Science and held various positions including Dean of Graduate Studies and Research, Dean of Institutional Relations, and Dean of Continuing Education and Community Service. Currently he is Assistant Professor at PMU. His research Interests include, Modelling and Realization of Biological Mechanism using CAD and FPGAs, Modelling and Realization of Arithmetic Operations using CAD and FPGAs, Low Power Cache Design, and Assistive Technology for students with disabilities.

Majid Ali Khan is working as Assistant professor at Prince Mohammad bin Fahd University, Saudi Arabia. He completed his PhD in Computer Engineering from University of Central Florida, USA, 2007.He earned his BS in Computer System Engineering, Ghulam Ishaq Khan (GIK) Institute of Engineering Sciences and Technology, Pakistan, 1997. His research interest includes Multi-agent Systems, Data Mining, Software Engineering.