Engineering and Technology Journal
Journal homepage: https://etj.uotechnology.edu.iq

Detection of arabic sign language by machine learning techniques with PCA and LDA

Mosab A. Hassan*[a], Atheer A. Sabri[b], Alaa H. Ali[a]

[a] Electrical Engineering Dept., University of Technology-Iraq, Alsina'a street, 10066 Baghdad, Iraq.
[b]Communications Engineering Dept., University of Technology-Iraq, Alsina'a street, 10066 Baghdad, Iraq.
*Corresponding author Email: eee.20.14@grad.uotechnology.edu.iq

HIGHLIGHTS

- An automatic system translated Arabic Sign Language to improve deaf-hearing communication.
- KNN algorithm had 86.4% accuracy in recognizing 32 Arabic sign language letters.
- The system bridges the communication gap between sign language users and non-users.

ARTICLE INFO

Handling editor: Ivan A. Hashim

Keywords: Arabic sign language, Machine Learning, K-Nearest Neighbor, Decision Tree, Naïve Bayes

ABSTRACT

People need a communication channel and a way to communicate with a person or group. Deaf people communicate with others through sign language. The remarkable and rapid development in image and video recognition systems has made researchers use this development to solve many problems, including sign language for the deaf, and reduce their suffering in communicating with ordinary people. This work aims to use and apply machine learning technology to build an automatic recognition system for Arabic Sign Language (ArSL). In this work, images of ArSL characters were recognized using four classification techniques (Naïve Bayes (NB), Decision Trees (DTs), and Adaptive Boosting), and K-Nearest Neighbor (KNN)) with the Python library and using two feature extraction algorithms (PCA & LDA). Data pre-processing steps, including grayscale conversion, Gaussian blur, histogram equalization, and resizing, are applied to enhance the data's suitability for training and testing. The work was tested with five experiments chosen with multiple ratios for training and test data. The first training is 90%, the second training is 80% of the data, the third is 75%, the fourth is 70%, and the last is 60%. The work also played a good role in interpreting ArSL, and the accuracy of the work considers the KNN algorithm more accurate in prediction.

1. Introduction

There are several ways of communication between humans, including speech, writing, hand gestures (visual language), drawing, and others [1]. The dominant mode of human communication is speech. When this is obstructed, humans must use a method of communication other than the above [2,3]. A percentage of the world's population suffers from their inability to speak (people who are deaf or hard of hearing), and their number is estimated in the millions [4,5]. This percentage is not small, and because they are an important part of society, it is important to solve the problem of people who are deaf or hard of hearing and reduce the communication gap between them and normal people [6]. Researchers used modern technology to find a way to communicate between deaf people and others. The dominant language among deaf people is manual sign language (gestures). A sign language that can be perceived visually, that is, with the naked eye. Sign language varies from country to country and has its own rules for that country [7,8]. There is no universal sign language.

Over 120 sign languages, such as American, British, Hindi, and Arabic, are used worldwide [9,10]. Gestures are divided into static language, which mostly represents letters and numbers, and dynamic, which represents words with video [11,12]. Sign language researchers collected information and data about these actions through two main methods: cameras and sensors represented by gloves and Microsoft Kinect V2 [13]. Gloves are difficult to use because they are heavy to wear, and the Microsoft Kinect V2 is too expensive [14]. The best and easiest way to collect data is through cameras. A system is designed to recognize Arabic sign language using machine learning on images, providing a solution for deaf people's integration into society. Sign

http://doi.org/10.30684/etj.2024.143688.1601
Received 08 October 2023; Received in revised form 12 January 2024; Accepted 16 January 2024; Available online 15 February 2024
2412-0758/University of Technology-Iraq, Baghdad, Iraq

language gesture segments and prediction of sign language dynamic movements, where the system overcame problems facing dynamic hand gestures, including segmentation, hand detection, feature representation, and classification.

2. Related works

Previously, much research has been presented and published on sign language in general. Below is some research related to Arabic sign language. In 2011, Youssif et al. developed an automatic recognition system for Arabic Sign Language ArSL based on Hidden Markov Models (HMMs). The identification of 20 isolated words in the Standard Arabic Sign Language was done using a large sample set. Use authentic ArSL recordings of deaf persons with various attire and skin tones. The total recognition rate for this system was 82.22% [15]. In 2014, Elons et al., developed a Graphics processing unit GPU-based program for Arabic sign language recognition that outperformed the initial version almost seven times. Multiplicative Neural Network MNN served as the classifier, while Pulse Coupled Neural Network PCNN was employed to extract the static mode features. Both training time and accuracy tests have demonstrated that MMNN is better than traditional Multilayer perceptron MLP. 83% of the 200 marks in the system were used to test the data [16]. In 2018, Rosero et al., used electronic gloves with sensors to obtain data representing numbers 1-9. They then applied Principal Component Analysis (PCA) to the data. In extracting the features, this system achieved a success rate of up to 85% by the K-nearest neighbor with the Constrained Horn Clause (KNN CHC) technique [17]. In 2018, Elpeltagy et al., presented a set of data consisting of 150 signs of Arabic sign language, and the total data used for these signs was 7500 images. This data was collected by Kinect v2, and HOG-PCA can extract features from the data. An accuracy of up to 55% was achieved by using the Random Forest classifier [18]. In 2018, ALZohairi et al., used a new ArSL recognition system. A mobile camera was used to collect data from 30 volunteers, collecting 900 images of the Arabic alphabet. The proposed system extracts the histogram of oriented gradients HOG descriptor from images, a Support vector machine SVM classifier. The resulting system succeeded in recognizing 63.5% of the gestures of the Arabic alphabet [19]. In 2021, Tharwat et al., proposed a system to recognize 14 Arabic sign language alphabet letters. Three data types and four classification algorithms were used. The KNN algorithm gave the highest accuracy of 99.5% [20].

In this research, our paper delves into the dynamic and promising domain dedicated to enhancing the lives of individuals with hearing impairments by utilizing cutting-edge machine learning and computer vision techniques to recognize Arabic Sign Language (ArSL). Our study underscores the diligent application of diverse classification methodologies and feature extraction algorithms. It is imperative for us to duly acknowledge the noteworthy prior research undertaken by Youssif et al. [15], Elons et al. [16], Rosero et al. [17], Elpeltagy et al. [18], ALZohairi et al. [19], and Tharwat et al. [20] in the domain of ArSL recognition.

In this paper, we endeavor to elucidate how our proposed system builds upon the accomplishments of these prior studies or offers innovative solutions to address existing limitations. By placing our work within the context of these advancements, we aim to underscore its significance in the broader landscape of ArSL recognition research. This contextualization is pivotal in illustrating how our approach plays a pivotal role in mitigating the communication barriers confronted by the deaf community, thereby amplifying their capacity to engage effectively with the world around them. Table 1 represents an overview of Related Works in Arabic Sign Language Recognition.

Table 1: Overview of Related Works in Arabic Sign Language Recognition

Year	Approach and Techniques	Dataset and Features	Recognition Rate	Ref.
2011	Hidden Markov Models (HMMs)	Authentic ArSL recordings, various attire, skin tones	82.22%	[15]
2014	GPU-based program, MNN classifier, PCNN	Standard Arabic Sign Language, static mode features	83%	[16]
2018	Electronic gloves with sensors, PCA	Data representing numbers 1-9	Up to 85%	[17]
2018	Kinect v2, HOG-PCA	150 signs of Arabic sign language, 7500 images	55.57%	[18]
2018	Mobile camera, HOG descriptor, SVM classifier	900 images of the Arabic alphabet	63.5%	[19]
2021	Multiple data types, KNN algorithm	14 Arabic sign language alphabet letters	99.5%	[20]

3. Proposed system

Sign language recognition is a vital technology that bridges the communication gap between deaf individuals and others. This section explains the processes and algorithms used in the proposed work to prepare and then classify the data.

The proposed Arabic Sign Language Alphabet Recognition System (ALARS) includes four main stages. The first stage begins with the data set used. Pre-processing includes (RGB to grayscale, Gaussian blur, Histogram equalization, and resizing), the second stage. The third stage is extracting features using two algorithms (PCA & LDA). Finally, it is the classification stage using four methods, which are (NB, DT, Ada-boosting, and KNN). This section provides a detailed explanation of each stage of the proposed system according to Figure 1.

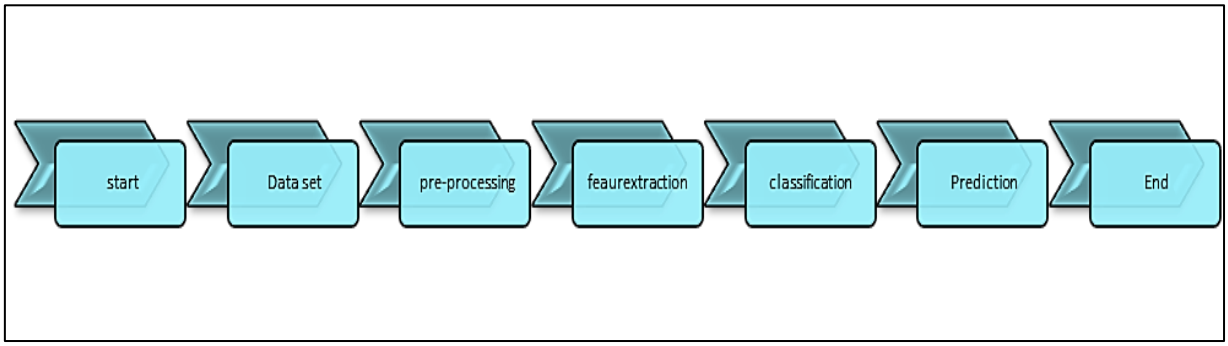

Figure 1: Proposed System

3.1 Data set stage

The data is the Arabic Sign Language alphabet used in this work. It consists of 54,049 grayscale images created by more than forty people of 32 and classical Arabic letters with different backgrounds. Lighting ArSL2018 is a novel, comprehensive, completely classified dataset of Arabic Sign Language images created at Prince Mohammed bin Fahd University in Al Khobar, Saudi Arabia, to be made available to deep learning and machine learning academics. It can be used to create helpful applications and devices for deaf and hard-of-hearing people. The ArSL 2018 dataset is unique because it is the first large comprehensive dataset for Arabic sign language. The dimensions of the images are 64*64 pixels [21] Figure 2 represents a sample of three letters of the Arabic Sign Language alphabet (Aleff (أ), Bba (ب), and Ain (ع)) and the operations that are performed on the letters before extracting their features. Table 2 represents the number of data images used in this work.

Aleef (أ) **Bba (ب)** **Ain (ي)**

Figure 2: Original Images [21]

Table 2: Arabic Alphabet Sign classes with a number of images [21]

No	Letter name in English Script	Letter name in Arabic scrip	No of Image	No	Letter name in English Script	Letter name in Arabic scrip	No of Image
1	Aleef	(أَلِف)أ	1672	17	Zā	(ظَاء)ظ	1723
2	Bba	(بَاء) ب	1791	18	Ayn	(عَين)ع	2114
3	Tā	(تَاء) ت	1838	19	Ghayn	(غَين)غ	1977
4	Thā	(ثَاء) ث	1766	20	Fā	(فَاء)ف	1955
5	Jīm	(جِيمْ) ج	1552	21	Qāf	(قَاف) ق	1705
6	Hā	(حَاء) ح	1526	22	Kāf	(كَاف)ك	1774
7	Khā	(خَاء) خ	1607	23	Lām	(لأمْ)ل	1832
8	Dāl	(دَالْ) د	1634	24	Mīm	(مِيمْ)م	1765
9	Dhāl	(ذَال) ذ	1582	25	Nūn	(نُون)ن	1819
10	Rā	(رَاء) ر	1659	26	Hā	(هَاء)ه	1592
11	Zāy	(زَايِ) ز	1374	27	Wāw	(وَاو)و	1371
12	Sīn	(سِينْ) س	1638	28	Yā	(يَا) ئ	1722
13	Shīn	(شِينْ) ش	1507	29	Tāa	(تَاء)ة	1791
14	Sād	(صَادْ)ص	1895	30	Al	(ال)ال	1343
15	Dād	(ضَاد)ض	1670	31	Laa	(لا)لا	1746
16	Tā	(طَاء)ط	1816	32	Yāa	(يَاء) يَاء	1293

3.2 Pre-processing

Operations performed on raw data for feature extraction and classification, which consists of four operations:

3.2.1 Convert grayscale

The main purpose of this process is to convert RGB (red, green, blue) color images to grayscale to reduce the data size, not the original information contained in the image, to help simplify the work of algorithms. Where they are The color image is 24-bit, while the grey image is 8-bit [22, 23]. The color image can be converted to grayscale according to Equation 1. A one-dimensional grayscale image matrix. Color intensity in the gray matrix indicates saturation and hue while ignoring luminance by maintaining color brightness with special weights based on (R, G, B) the color components in the real image, where the color components are weighted according to the system's formula—colors transition from RGB to grayscale. To finish the job, we included this step in the case of colored data. Employ the weighted method because it produces better results than the average method. After all, the standard method produces blurry images.

$$\mathrm{GREY} = 0.30\ \mathrm{R} + 0.59\ \mathrm{G} + 0.11\ \mathrm{B} \tag{1}$$

Grayscale images were used in the work because most image processing processes treat grayscale images like filters, expressing image information as a single component that indicates the intensity of color in the image while preserving image details.

In the context of Equation 1, it is imperative to elucidate the choice of weight values for converting color images to grayscale. These weight values are not arbitrary but are based on the human visual system's sensitivity to different color channels. For example, the human eye is more sensitive to changes in green colors than red and blue. Therefore, assigning a higher weight to green (0.59) in the grayscale conversion formula acknowledges this heightened sensitivity and ensures that green color variations significantly impact the resulting grayscale image. Similarly, the lower weights for red (0.30) and blue (0.11) reflect their relatively lower importance in capturing color information while preserving image details in grayscale.

To illustrate this concept, consider a color image with a vibrant green grass field and a blue sky. The higher weight assigned to green in Equation 1 ensures that the varying shades of green in the grass will be more prominently represented in the grayscale version, maintaining the visual emphasis on the grass's details. Meanwhile, the lower weights for red and blue contribute to a more muted representation of the blue sky and any red objects in the image, as these colors are less critical for conveying detailed information in grayscale.

In essence, these weight values in the grayscale conversion formula are carefully chosen to align with human perception, ensuring that the resulting grayscale images effectively convey essential image information while simplifying data for computational processing.

3.2.2 Gaussian blur

The standard deviation of the Gaussian distribution is used to calculate the modification to apply to each pixel in the picture when using the Gaussian blur, a sort of image-blurring filter, to describe the normal distribution in statistics. σ is the standard deviation of the Gaussian distribution [24].

$$G(q) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-q}{2\sigma^2}} \tag{2}$$

G(q) represents the Gaussian blur, and σ is the standard deviation of the Gaussian distribution. In this context, σ essentially acts as a tuning parameter, allowing us to control the level of smoothing or blurring applied to the image.

To ensure clarity and transparency in our paper, we acknowledge the importance of explaining the parameters used in Equation 2. We recognize that a thorough understanding of these parameters is essential for readers to grasp the underlying principles and nuances of the Gaussian blur technique as applied in our research. By elucidating the significance of σ and its role in image processing, we aim to provide readers with a comprehensive understanding of our methodology and its impact on image enhancement within the context of our study.

3.2.3 Histogram equalization

Image quality can be improved using the common approach of histogram equalization. This process is very similar to histogram stretching, although it often yields more attractive results on larger images. This approach preserves the overall shape of the graph while producing a final image with a histogram that is as flat as realistically achievable [25]. To enhance the contrast of an image, the histogram can be adjusted to make darker pixels appear darker and brighter pixels appear lighter, regardless of whether the black pixels are already dark enough. This results in a slight increase in brightness for dark pixels and a significant increase for bright pixels. The procedure for histogram equalization in digital images involves four steps, as outlined in [26].

Step 1: Calculate the cumulative graph values.
Step 2: To normalize the values from Step 1, divide them by the total number of pixels.
Step 3: Multiply the values in Step 2 by the larger grey level and round to the nearest integer.
Step 4: The grey level values must be precisely aligned with the outcomes derived from Step 3.

The cumulative distribution function of a histogram is crucial for computing histogram equalization, as demonstrated in Equations 3 and 4:

$$\mathrm{Cdf\,(R)} \;=\; \sum_{i=1}^{r} \mathrm{h(i)} \tag{3}$$

R represents the grey value, and h illustrates the mage's histogram.

$$\mathrm{T[pixel]} \;=\; \mathrm{round}\,(\,(\,\frac{\mathrm{cdf(r)-\,cdf(r)_{min}}}{\mathrm{E*F-cdf(r)_{min}}}\,)\;*\;(\mathrm{L}-\,1) \tag{4}$$

The minimum value of the cumulative distribution function is Cdf (r)_min, where E and F represent the number of columns and rows of images, respectively, and L represents the number of grey levels used, which is 256.

3.2.4 Resizing

An important stage in changing the dimensions of the image in this work is to reduce the dimensions of the image while preserving the original information of the image so as not to take up large storage space and the ease of training and classification using bilinear interpolation Equation (5) [27,28] :

$$y = y_o(1 + \frac{x-x_o}{x1-x_o}) + y1(1 - \frac{x^1-x}{x1-x_o}) \tag{5}$$

Starting at (x, y1) and (x, y2), do two linear interpolations in the x-direction (horizontal). Use the interpolated values at (x, y1) and (x, y2) to calculate the interpolation at the final location (x, y), and then execute linear interpolation in the y-direction (vertical). The values of the pixels used to create interpolar objects will then be adjusted appropriately. The interpolated y value is shown between pixels x_o and x_1, and vice versa for pixels y_o and y_1. The y-values cannot be greater than the x_o, x_1, y_o, and y_1 values, so keep that in mind. This equation yields the color scheme of a camera. This equation produces the color scheme for the camera. The bipolar line coefficient depends on the distances between nearby pixels in horizontal and vertical directions, revealing an image composition of white and black color patterns separated by grayscale transitions.

3.3 Feature extraction

Feature extraction aims to facilitate pattern classification by obtaining more information from the data to be classified and representing it with lower-dimensional data without losing the original information. This work will use two methods to extract features into PCA and LDA categories.

3.3.1 Principal component analysis (PCA)

A statistical algorithm converts related variables into uncorrelated ones [29]. The goal of PCA is to describe the pattern with smaller amounts of features and thus reduce the dimensionality of the feature space while retaining the original information by the two Equations (6 and 7) below [30,31].

$$\mathrm{Average} = \frac{1}{\mathrm{M}}\sum_{n=1}^{\mu} \mathrm{Training\ images\ (n)} \tag{6}$$

$$\mathrm{Covriance} = \sum_{n=1}^{\mu} \mathrm{sub(n)\ sub^T(n)} \tag{7}$$

where M is the total images training set, Sub represents the subtracted image from the average μ, and μ: represents the average Mean.

3.3.2 Linear discriminant analysis (LDA)

One of the uses of LDA is pattern recognition and computer applications because it works as a traditional statistical method that reduces the image's dimensions while preserving the necessary information for recognition [32]. LDA aims to identify a matrix that maximizes the separation between class feature vectors in a low-dimensional space. For vectors, the attributes of other categories are given in [33]. The work of this method is explained by the following Equations (8-13):

$$\mu_j = \frac{1}{n_j}\sum_{x_i \in \omega_j} x_i \tag{8}$$

$$\mu = \frac{1}{\mathrm{N}}\sum_{i=1}^{\mathrm{N}} \mathrm{x_i} = \sum_{i=1}^{\mathrm{c}} \frac{\mathrm{n_i}}{\mathrm{N}}\mu_i \tag{9}$$

$$S_B = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{10}$$

$$S_W = \sum_{j=1}^{c}\sum_{i=1}^{n_j}(x_{ij} - \mu_j)(x_{ij} - \mu_j)^T \tag{11}$$

where x_i Represents the ith sample in the jth class.

$$W = S_W^{-1} S_B \tag{12}$$

$$Y = XV_K \tag{13}$$

where: ni represents the number of samples of the ith class, N is the total number of samples, μ Projection of the total Mean of all classes μi represents the projection of the Mean of the ith, SWi is the within-class variance of the ith class, SB between-class variance represents the difference between the Mean.

3.4 Classification

Analytical models for analyzing data can be created automatically using machine learning, a subset of software built on the idea that computers can learn independently from data, see patterns, and draw conclusions with a little help from humans. The following sections provide more information about the four algorithms used in this study: NB, DT, Ada Boost, and KNN.

3.4.1 Naïve bayes

Algorithm NB is based on probability by defining a set of probabilities. Probabilities can be defined by calculating the frequency and sets of values in a specific data set [34]. The value of the variable class applies the Bayesian method and assumes that all attributes are independent. The NB method is good and quickly captures new information from the positions of the supervised classification. Algorithm NB uses the following Equation (14) [35]:

$$P\left(\frac{A}{B}\right) = \frac{P\left(\frac{B}{A}\right)P(A)}{P(B)} \tag{14}$$

P(A/B) = Posterior probability, P(A) = Prior probability, P(B/A) = Likelihood, P(B) = Marginal probability.

3.4.2 Decision tree

It is a non-parametric and supervised learning technique used in classification and regression [36]. By learning direct decision rules inferred from data attributes, the goal is to develop a model that predicts the value of the target variable. A decision tree can be viewed as a multi-definition constant approximation, and it is possible to classify it using the Equation (15) :

$$E(S) = -\sum_{i=1}^{k} Pi\ Logk(Pi) \tag{15}$$

E(S) = Current state, Pi = Probability of an event i of state S or percentage of class i in the anode of state S, k = number of states.

3.4.3 Adaptive boosting

Classification results are improved using adaptive augmented ML technology. Descriptive algorithms can enhance performance when combined with other learning algorithms. It responds to incorrectly categorized examples by changing subsequent classifiers. Thus, ADA uses a weak classifier and adjusts the weight of each sample with each call [37]. Incorrectly classified samples will be given more weight in this procedure than correctly classified samples, resulting in them being favored by the new classifier. Algorithm Ada Bossting uses the following Equation (16) [38, 39]:

$$H(X) = sign(\sum_{t=1}^{T} \alpha tht(x)) \tag{16}$$

T= weak classifier, h(x) =output of weak classifier "t", α = weight applied to classifier "t."

3.4.4 K-Nearest Neighbors

A supervised machine learning algorithm is used. It is a supervised learning algorithm and one of the most well-liked instance-based learning methods for pattern recognition [40]. K-NN works well when all the data have the same scale and is a lazy learner since it lacks a training step. Because of its ease of use, the KNN idea has become a popular technique for categorization across various applications. For instance, the algorithm first looks for its K nearest neighbors in the feature space to categorize a sample Si using the feature vectors and the specified distance. After that, the algorithm executes the votes of these neighbors by their labels [41]. The group with the most neighbors with the same label will be assigned to the object sample [42]. Multiple distance metrics are used in the K-Nearest Neighbor (KNN) algorithm. In this work, the best measure is the Euclidean distance. The normal straight-line distance (Euclidean distance) between two points (x, y). Euclidean distance Equation (17):

$$\text{Euclidean distance} = \sqrt[2]{\sum_{i=1}^{n}(xi - yi)^2} \tag{17}$$

4. Results and discussion

This section presents the processes and results of the experiments conducted in this study and interpretations of these results in light of the study objectives. The ideal precision value is the most efficient value for testing the performance of four algorithms. The value is generated when applied to the training and test dataset sizes. Four classification methods were used, and at this

stage, the work was tested to determine its accuracy in distinguishing gesture movement and predicting the correct letter choice for 32 Arabic letters. The work will be tested in five experiments based on the proportion of trained and tested data for 54,049 images to know the effect of the number of trained data on this job. The Figure 3 represents the operations performed in this work, and we will explain the operation of each step. First, we will display the images resulting from the previous operations and then the accuracy results of this work.

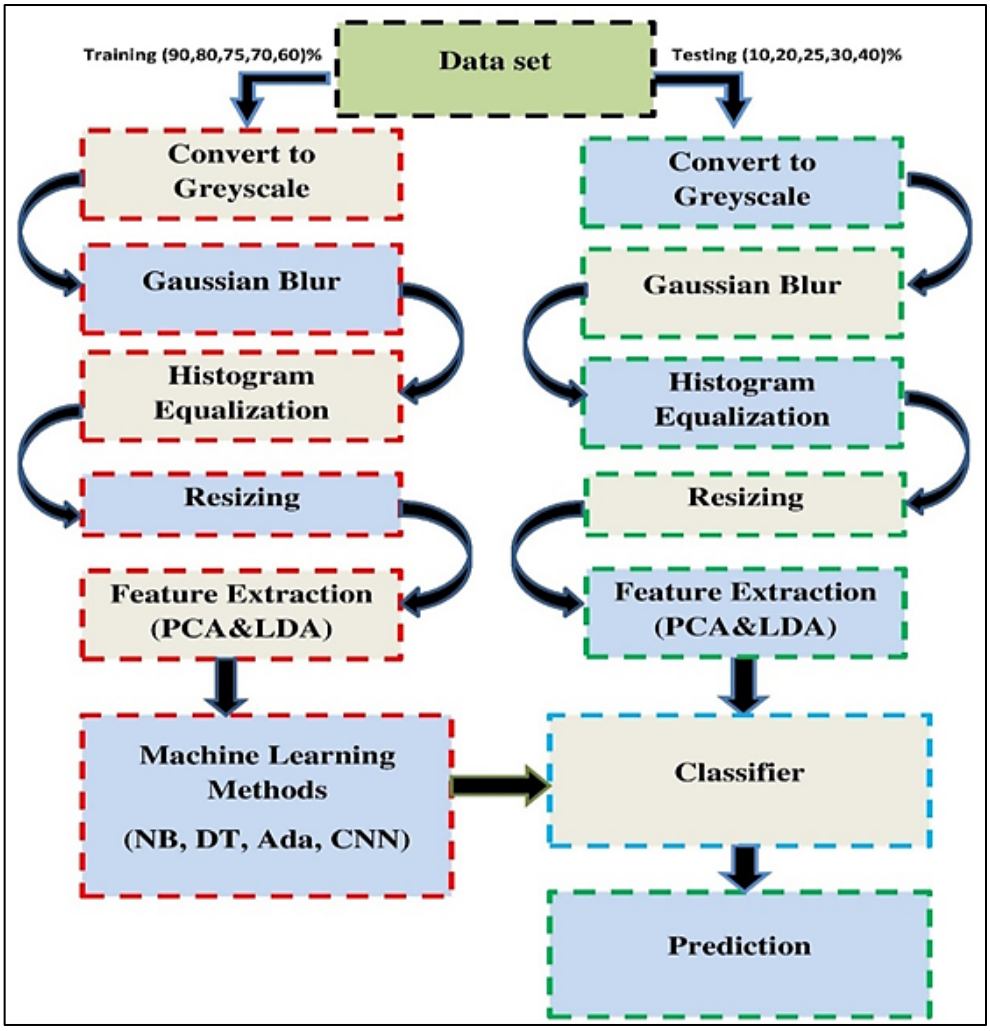

Figure 3: Suggested system

4.1 Pre-processing

It includes four operations to improve and prepare images, which are in sequence:

4.1.1 Grayscale

It is a very important process because it reduces the colors (red, green, and blue) to greyscale, and this is useful for a smaller data space and ease in other operations because color images need several operations to be dealt with. From Equation (1), it is converted from colors to grayscale, and the result appears in Figure 4.

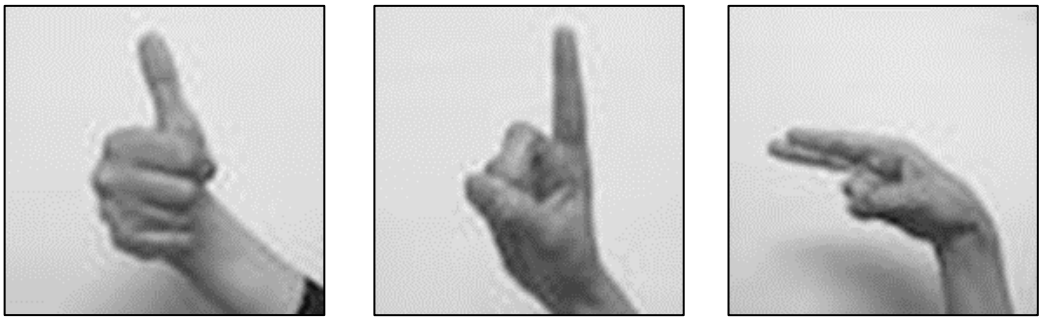

Figure 4: Grayscale Images [21]

4.1.2 Gaussian blur

A necessary process to remove noise from the image and improve it, according to Equation (2), and Figure 5 shows the image after filtering.

Figure 5: Gaussian Blur Images

4.1.3 Histogram equalization

Treatment of the difference in lighting and backgrounds of the images is done according to Equations 3 & 4, and Figure 6 shows the image after the procedure.

Figure 6: Histogram Equalization Images

4.1.4 Resizing

To reduce image size and storage space, use reduction Equation (5). Final images will be 20x20 pixels, as shown in Figure 7.

Figure 7: Resizing Images

4.2 Feature extraction

LDA and PCA are two commonly used machine learning techniques for dimensionality reduction and feature extraction. The Linear Discriminant Analysis (LDA) aims to find an efficient way to represent the gesture vector space. PCA constructs the gesture space using the whole gesture training data and not the gesture class information.

4.3 Classification

The data is divided into training data and testing data. This work was divided into five experiments. In the first experiment (90% of the data was subjected to training, and the remainder of the images were tested). The second experiment (80% of the data was subjected to training and the remainder for testing). In the third experiment (75% of the data was subjected to training and the remainder to testing). The fourth experiment (70% of the data was subjected to training and the remainder to testing) The fifth experiment (60% of the data was subjected to training and the remainder to testing). The five experiments were applied to the four algorithms, and the KNN algorithm showed the best classification. The five experiments aim to determine the change in accuracy, and it turns out that the more training data there is, the better the accuracy.

The first experiment divided the data into 90% for training and 10% for testing. The test was carried out using four classification methods (Ada boosting, DT, NB, and KNN), and it is shown in Figure 8 that KNN has the highest classification accuracy at 86.4%.

The second experiment divided the data into 80% for training and 20% for testing. The test was carried out using four classification methods (Ada boosting, DT, NB, and KNN), and it is shown in Figure 9 that KNN has the highest classification accuracy at 85.3%. The third experiment divided the data into 75% for training and 25% for testing. The test was carried out using four classification methods (Ada boosting, DT, NB, and KNN), and it is shown in Figure 10 that KNN has the highest classification accuracy, 84.6%.

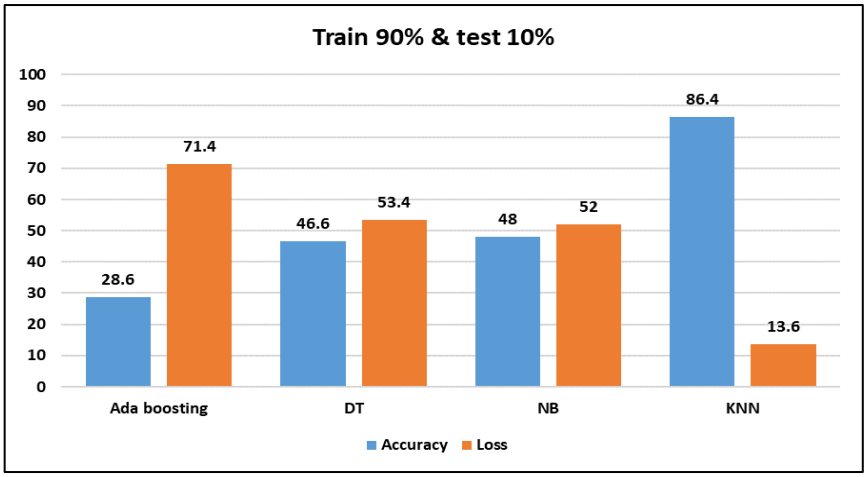

Figure 8: (Train Data 90% &Test Data 10%)

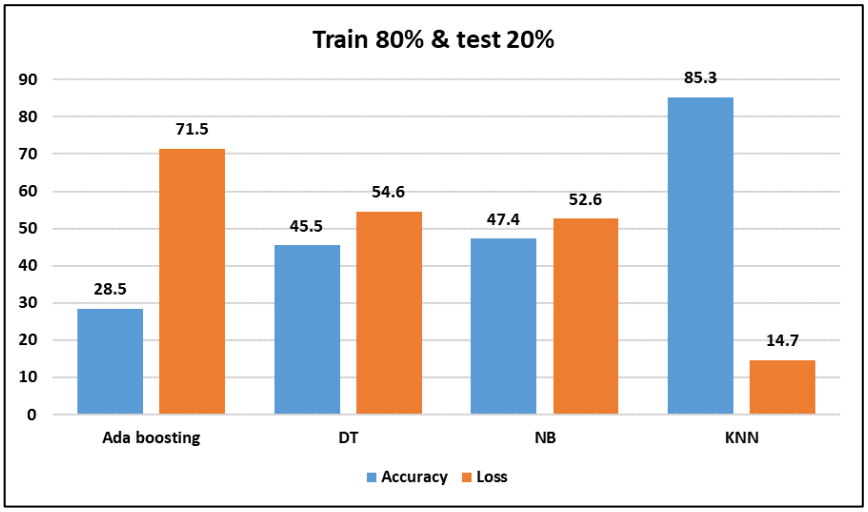

Figure 9: (Train Data 80% &Test Data 20%)

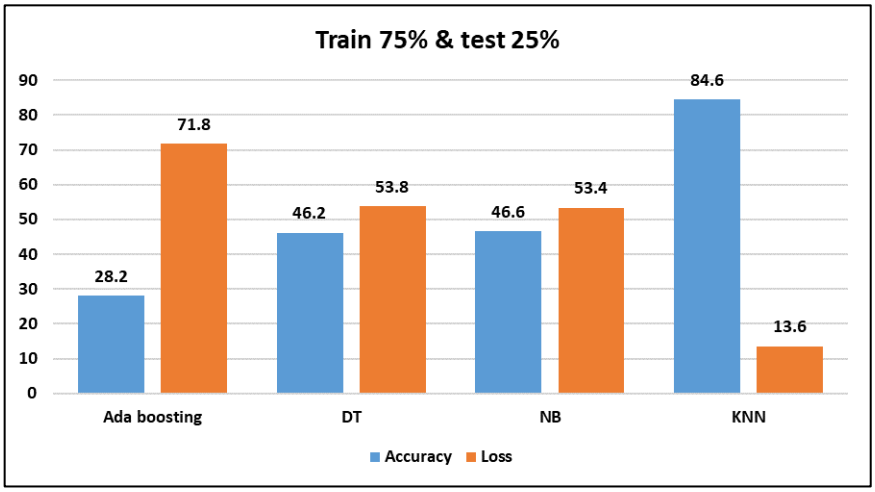

Figure 10: (Train Data 75% & Test Data 25%)

The fourth experiment divided the data into 70% for training and 30% for testing. The test was carried out using four classification methods (Ada boosting, DT, NB, and KNN), and it is shown in Figure 11 that KNN has the highest classification accuracy, 84.1%. The Fifth experiment divided the data into 60% for training and 40% for testing. The test was carried out using four classification methods (Ada boosting, DT, NB, and KNN), and it is shown in Figure 12 that KNN has the highest classification accuracy, 82.9%.

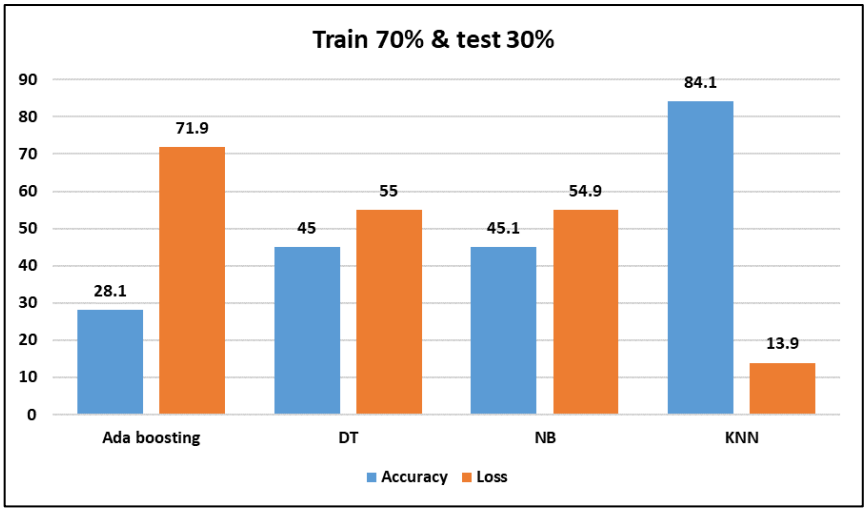

Figure 11: (Train Data 70% & Test Data 30%)

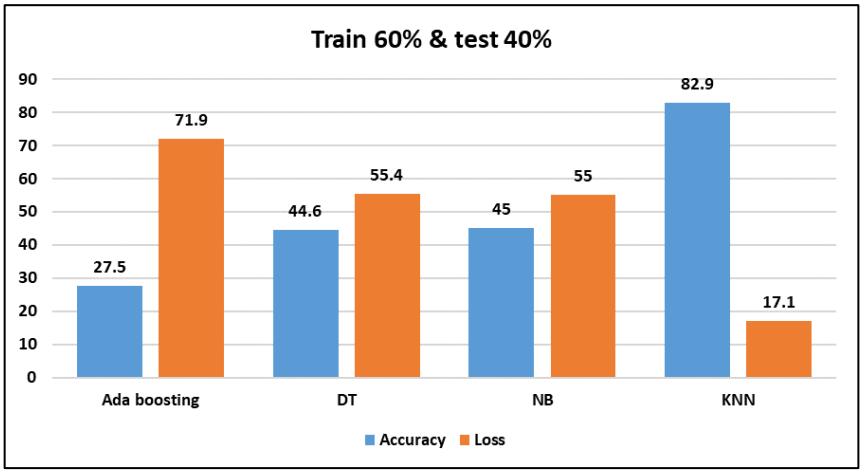

Figure 12: (Train Data 60% & Test Data 40%)

The experiments (1, 2, 3, 4, and 5) show the performance of the four algorithms. Notice that the algorithm Ada Boosting is less accurate than the other algorithms because its work aims to build a model on a data set and then build a second model to correct the errors present in the first model. This procedure continues until the errors are reduced to a minimum and a data set is predicted. Correctly, it combines multiple model weak learners to arrive at the final results of strong learners. Therefore, this algorithm increases the variance of the model step by step across iterations. Hence, this algorithm gives less accuracy than other algorithms, and these reasons make this algorithm weak in classification for such data.

The decision tree algorithm visually displays decisions, their results, consequences, and potential costs. Thus, branches can easily be evaluated and compared to choose the best courses of action, i.e., it predicts the value of the target variable by learning simple decision rules inferred from the features of the data. This algorithm applies a top-down approach to a set. The data is fed during training, and the decision tree classifies the data without requiring many calculations, feature reduction, and data re-sampling. These are the reasons that this algorithm is less than KNN. A decision tree is superior to ada boosting because decision trunks are decision trees with one node and two leaves. AdaBoost employs many decision processes, each dependent on a single variable or feature. In contrast to the random forest, decision trees employ numerous factors to determine the final categorization choice in decision trees.

Algorithm Naïve Bayes is a statistical classifier that relies on the principle of probability. The algorithm seeks to model the distribution of the inputs of a particular class or class, where all predictors are assumed to be independent, which rarely happens in real life. You should not take its probabilistic outputs seriously because its estimates can sometimes be off. The biggest drawbacks of this algorithm are the requirements for forecasters to be independent, and it is also weak at classifying large data. Regarding computation, Nave Bayes outperforms Ada boosting and Decision Tree in learning and classification tasks.

KNN achieved the best results because it relies on similarities in observable data and sophisticated distance metrics to generate accurate data, performs the calculations required for each step, and scans the entire data set for prediction. After all, it does not generalize the data in advance.

Table 3 compares the proposed and previous work [15-20]. The work of Youssef et al. [15] was 82.22% better than when sign language was highlighted. However, the research faced the problem of poor image optimization, especially lighting, before classification. On the other hand, we worked on improving the images before classification and using the histogram equalization method to solve the image lighting problem. Elson et al. [16], the time taken to perform the system was very large, about 15.55 hours to distinguish 200 signals, while our work for four algorithms took 20 seconds. Risero et al. [17] Gloves were used to collect data, and the number of images was 5000. The KNN algorithm took 32 minutes, while the proposed system was able to discriminate using the same algorithm in less time. Elbitaje et al. [18] The features extracted from the images are greater than those extracted from the images of the proposed system. In other words, our system works with fewer features and with higher accuracy. Al-Zuhairi et al. [19] Some letters had an accuracy of zero, more than four letters of the Arabic Sign Language alphabet. At the same time, in our proposed system there is no accuracy equal to zero for any letter, and the last letter obtained an accuracy of 26%. Tharwat et al. [20] focused on recognizing 14 Arabic letters using three types of data with 99.5% accuracy. As for the source of [43], the work took 66 a minute, and this time is much more than the time used in our work, as our work took about 20 minutes. In comparison, our work was able to distinguish 32 Arabic letters with a lower accuracy of 86.4%. Our work is distinguished by the ability to distinguish 32 letters instead of 14 letters.

Table 3: Comparing Previous Works

No	Input data method	Extraction feature method	Classifier method	Output type	Accuracy %	Ref.
1	Camera	Feature extraction method	HMM	20 words	82.22	[15]
2	Camera	PCNN	MMNN	200 signs	83	[16]
3	Glove	PCA	KNN -CHC	Number (1-9)	85	[17]
4	Camera	HGO-PCA	Random forest	150 signs	55	[18]
5	Camera	HGO	SVM	30 letters	63.5	[19]
6	Camera	Based on shape	KNN	14 letters	99.5	[20]
7	Camera	-	ArSL-CNN	32 letters	97.29	[43]
8	Camera	PCA&LDA	NB	32 letters	46.6	Current proposed
			DT		47.4	
			ADA		28.5	
			Boosting		86.4	
			KNN			

5. Conclusion

This paper describes an innovative method to improve communication between deaf and hearing people – an automatic translation system for Arabic Sign Language. This system can help people who are deaf or hard of hearing in daily life and help others learn sign language. This work investigated four different machine-learning algorithms. This study aimed to distinguish 32 letters of the Arabic sign language alphabet. By comparing the results between these algorithms, it was found that the KNN algorithm is the best among the other algorithms while using the same data for all algorithms. The research concluded that the KNN model has higher accuracy when predicting hand gesture accuracy (86.4%) using data indicating that different machine learning methods produce such a wide range of results. The system's primary goal is to bridge the communication gap between those who use sign language and those who do not. The system can be extended to recognize phrases and words.

Author contributions

Conceptualization, M. Hassan., A. Sabri and A. Ali; software, M. Hassan., A. Sabri and A. Ali.; formal analysis, M. Hassan M. Hassan; resources, A. Sabri.; data curation, A. Ali.; writing—original draft preparation, M. Hassan.; writing—review and editing, M. Hassan.; supervision, A. Sabri and A. Ali.; project administration, M. Hassan M. Hassan. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data availability statement

The data that support the findings of this study are available on request from the corresponding author.

Conflicts of interest

The authors declare that there is no conflict of interest.

References

[1] Alnahhas, B. Alkhatib, N. Al-Boukaee, N. Alhakim, O. Alzabibi, and N. Ajalyakeen, Enhancing the recognition of Arabic sign language by using deep learning and leap motion controller, Int. J. Sci. Technol. Res, 9 (2020) 1865–1870.

[2] A. Das, S. Gawde, K. Suratwala, and D. Kalbande, Sign language recognition using deep learning on custom processed static gesture images, Int. Conf. Smart City Emerg. Technol., (2018) 1–6. https://doi.org/10.1109/ICSCET.2018.8537248

[3] V. Jain, A. Jain, A. Chauhan, S. S. Kotla, and A. Gautam, American sign language recognition using support vector machine and convolutional neural network, Int. J. Inf. Technol., 13 (2021) 1193–1200.

[4] H. N. Saha, S. Tapadar, S. Ray, S. K. Chatterjee, and S. Saha, A machine learning based approach for hand gesture recognition using distinctive feature extraction, in 2018 IEEE 8th Annual Computing and Communication Workshop and Conference, (2018) 91–98. https://doi.org/10.1109/CCWC.2018.8301631

[5] N. Krishnaraj, M. G. Kavitha, T. Jayasankar, and K. V. Kumar, A Glove based approach to recognize Indian Sign Languages, Int. J. Recent Technol. Eng. IJRTE, 7 (2019) 1419–1425.

[6] R. M. Mohammed and S. M. Kadhem, A review on Arabic sign language translator systems, J. Phys. Conf. Ser., 1818 (2021) 12033. https://doi.org/10.1088/1742-6596/1818/1/012033

[7] N. Aloysius and M. Geetha, Understanding vision-based continuous sign language recognition, Multimed. Tools Appl., 79 (2020) 22177– 22209. https://doi.org/10.1007/s11042-020-08961-z

[8] S. Alshomrani, L. Aljoudi, and M. Arif, Arabic and American sign languages alphabet recognition by convolutional neural network, Adv. Sci. Technol. Res. J., 15 (2021) 136 –148. https://doi.org/10.12913/22998624/142012

[9] J. Huang, W. Zhou, H. Li, and W. Li, Attention-based 3D-CNNs for large-vocabulary sign language recognition, Trans. Circuits Syst. Video Technol., 29 (2019) 2822–2832. https://doi.org/10.1109/TCSVT.2018.2870740

[10] H. Hameed et al., Privacy-preserving British sign language recognition using deep learning, 44th Annual International Conference of the Eng. Med. Biol. Soc., (2022) 4316 – 4319. https://doi.org/10.36227/techrxiv.19170257.v1

[11] M. AL-Muifraje, T. Saeed, and M. Hassan, Design and Analysis of American Sign Language Classifier Based on n-tuple Technique, Eng. Technol. J., 36 (2018) 172–178. https://doi.org/10.30684/etj.36.2A.8

[12] A. H. Alrubayi et al., A pattern recognition model for static gestures in Malaysian sign language based on machine learning techniques, Comput. Electr. Eng., 95 (2021) 107383. https://doi.org/10.1016/j.compeleceng.2021.107383

[13] A. Kasapbaşi, A. E. A. Elbushra, A.-H. Omar, and A. Yilmaz, DeepASLR: A CNN based human-computer interface for American Sign Language recognition for hearing-impaired individuals, Comput. Methods Programs Biomed. Update., 2 (2022) 100048. https://doi.org/10.1016/j.cmpbup.2021.100048

[14] P. R. Sanmitra, V. V. S. Sowmya, and K. Lalithanjana, Machine Learning Based Real Time Sign Language Detection, Int. J. Res. Eng. Sci. Manag., 4 (2021) 137–141.

[15] A. A. A. Youssif, A. E. Aboutabl, and H. H. Ali, Arabic sign language (arsl) recognition system using hmm, Int. J. Adv. Comput. Sci. Appl., 2 (2011). https://dx.doi.org/10.14569/IJACSA.2011.021108

[16] A. S. Elons, GPU implementation for Arabic sign language real-time recognition using multi-level multiplicative neural networks, Int. Conf. Comput. Eng. Syst., (2014) 360 –367. https://doi.org/10.1109/ICCES.2014.7030986

[17] P. D. Rosero-Montalvo et al., Sign language recognition based on intelligent glove using machine learning techniques, Third Ecuador Technical Chapters Meeting (ETCM), 2018, 1–5. https://dx.doi.org/10.1109/ETCM.2018.8580268

[18] M. Elpeltagy, M. Abdelwahab, M. E. Hussein, A. Shoukry, A. Shoala, and M. Galal, Multi-modality-based Arabic sign language recognition, IET Comput. Vis., 12 (2018) 1031–1039. https://doi.org/10.1049/iet-cvi.2017.0598

[19] R. Alzohairi, R. Alghonaim, W. Alshehri, and S. Aloqeely, Image-based Arabic sign language recognition system, Int. J. Adv. Comput. Sci. Appl., 9 (2018). http://dx.doi.org/10.14569/IJACSA.2018.090327

[20] G. Tharwat, A. M. Ahmed, and B. Bouallegue, Arabic sign language recognition system for alphabets using machine learning techniques, J. Electr. Comput. Eng., 2021 (2021) 1–17. https://doi.org/10.1155/2021/2995851

[21] G. Latif, N. Mohammad, J. Alghazo, R. AlKhalaf, and R. AlKhalaf, ArASL: Arabic alphabets sign language dataset, Data Br., 23 (2019) 103777. https://doi.org/10.1016/j.dib.2019.103777

[22] S. A. H. Alrubaie and A. H. Hameed, Dynamic weights equations for converting grayscale image to RGB image, J. Univ. Babylon Pure Appl. Sci., 26 (2019) 122–129. https://doi.org/10.29196/jubpas.v26i8.1677

[23] C. Panigrahy, A. Seal, N. K. Mahato, and D. Bhattacharjee, Differential box-counting methods for estimating fractal dimension of greyscale images: A survey, Chaos, Solitons & Fractals, 126 (2019) 178–202. https://doi.org/10.1016/j.chaos.2019.06.007

[24] J. Flusser, S. Farokhi, C. Höschl, T. Suk, B. Zitova, and M. Pedone, Recognition of images degraded by Gaussian blur, IEEE Trans. Image Process., 25 (2015) 790 – 806. https://doi.org/10.1109/TIP.2015.2512108

[25] S. Patel and M. Goswami, Comparative analysis of Histogram Equalization techniques, International Conference on Contemporary Computing and Informatics, (2014) 167–168. https://doi.org/10.1109/IC3I.2014.7019808

[26] S. Dubey and M. Dixit, Image Enhancement Techniques: An Exhaustive Review, Int. Conf. Sust. Innov. Sol. Current Chall. Eng. Technol., (2019) 363 – 375. http://dx.doi.org/10.1007/978-3-030-44758-8_34

[27] P. S. Prasanna and P. V Virparia, A comparative analysis of image interpolation algorithms, Int. J. Adv. Res. Comput. Commun. Eng., 5 (2016) 29 – 34. http://dx.doi.org/10.17148/IJARCCE.2016.5107

[28] I. W. Adiyasa, A. P. Prasetyono, A. Yudianto, P. P. W. Begawan, and D. Sultantyo, Bilinear interpolation method on 8x8 pixel thermal camera for temperature instrument of combustion engine, J. Phys. Conf. Ser., 1700 (2020) 12076. http://dx.doi.org/10.1088/1742-6596/1700/1/012076

[29] H. M. Ebied, Feature extraction using PCA and Kernel-PCA for face recognition, 2012 8th International Conference on Informatics and Systems (INFOS), 2012, MM–72.

[30] W. Aly, S. Aly, and S. Almotairi, User-independent American sign language alphabet recognition based on depth image and PCANet features, IEEE Access, 7 (2019) 123138 –123150. http://dx.doi.org/10.1109/ACCESS.2019.2938829

[31] T. M. Angona et al., Automated Bangla sign language translation system for alphabets by means of MobileNet, TELKOMNIKA (Telecommunication Comput. Electron. Control., 18 (2020) 1292–1301. http://dx.doi.org/10.12928/TELKOMNIKA.v18i3.15311

[32] A. Sharma and K. K. Paliwal, Linear discriminant analysis for the small sample size problem: an overview, Int. J. Mach. Learn. Cybern., 6 (2015) 443–454. https://doi.org/10.1007/s13042-013-0226-9

[33] M. Deriche, S. O. Aliyu, and M. Mohandes, An intelligent Arabic sign language recognition system using a pair of LMCs with GMM based classification, IEEE Sens. J., 19 (2019) 8067–8078. https://doi.org/10.1109/JSEN.2019.2917525

[34] B. Hisham and A. Hamouda, Supervised learning classifiers for Arabic gestures recognition using Kinect V2, SN Appl. Sci., 1 (2019) 1–21. https://doi.org/10.1007/s42452-019-0771-2

[35] D. Silahudin and A. Holidin, Model expert system for diagnosis of COVID-19 using naïve Bayes classifier, IOP Conf. Ser. Mater. Sci. Eng., 1007 (2020) 12067. https://doi.org/10.1088/1757-899X/1007/1/012067

[36] A. Cherfi, K. Nouira, and A. Ferchichi, Very fast C4. 5 decision tree algorithm, Appl. Artif. Intell., 32 (2018) 119–137. https://doi.org/10.1080/08839514.2018.1447479

[37] B. Hisham and A. Hamouda, Arabic sign language recognition using Ada-Boosting based on a leap motion controller, Int. J. Inf. Technol., 13 (2021) 1221–1234. https://doi.org/10.1007/s41870-020-00518-5

[38] A. Taherkhani, G. Cosma, and T. M. McGinnity, AdaBoost-CNN: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning, Neurocomputing, 404 (2020) 351–366. https://doi.org/10.1016/j.neucom.2020.03.064

[39] A. Shahraki, M. Abbasi, and Ø. Haugen, Boosting algorithms for network intrusion detection: A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost, Eng. Appl. Artif. Intell., 94 (2020) 103770. https://doi.org/10.1016/j.engappai.2020.103770

[40] F. Utaminingrum, I. K. Somawirata, and G. D. Naviri, Alphabet Sign Language Recognition Using K-Nearest Neighbor Optimization., J. Comput., 14 (2019) 63–70. https://doi.org/10.17706/jcp.14.1.63-70

[41] N. K. Hayder and A. L. Behadili, Classification Algorithms for Determining Handwritten Digit, Electr. Electron. Eng., 12 (2016) 96–102. https://doi.org/10.37917/ijeee.12.1.10

[42] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, Efficient kNN classification with different numbers of nearest neighbours, IEEE Trans. Neural networks Learn. Syst., 29 (2017) 1774 –1785. https://doi.org/10.1109/TNNLS.2017.2673241

[43] A. A. Alani, & Cosma, G., ArSL-CNN: a convolutional neural network for Arabic sign language gesture recognition. Indones. J. Electr. Eng. Comput. Sci., 22 (2021). http://doi.org/10.11591/ijeecs.v22.i2.pp1096-1107