Computers, Materials & Continua
DOI: 10.32604/cmc.2022.023591
Article

Tech Science Press

Robust Interactive Method for Hand Gestures Recognition Using Machine Learning

Amal Abdullah Mohammed Alteaimi[1,*] **and Mohamed Tahar Ben Othman**[1,2]

[1]Department of Computer Science, College of Computer, Qassim University, Buraydah, 51452, Saudi Arabia
[2]BIND Research Group, College of Computer, Qassim University, Buraydah, 51452, Saudi Arabia
*Corresponding Author: Amal Abdullah Mohammed Alteaimi. Email: a.alteaimi@qu.edu.sa
Received: 13 September 2021; Accepted: 15 December 2021

Abstract: The Hand Gestures Recognition (HGR) System can be employed to facilitate communication between humans and computers instead of using special input and output devices. These devices may complicate communication with computers especially for people with disabilities. Hand gestures can be defined as a natural human-to-human communication method, which also can be used in human-computer interaction. Many researchers developed various techniques and methods that aimed to understand and recognize specific hand gestures by employing one or two machine learning algorithms with a reasonable accuracy. This work aims to develop a powerful hand gesture recognition model with a 100% recognition rate. We proposed an ensemble classification model that combines the most powerful machine learning classifiers to obtain diversity and improve accuracy. The majority voting method was used to aggregate accuracies produced by each classifier and get the final classification result. Our model was trained using a self-constructed dataset containing 1600 images of ten different hand gestures. The employing of canny's edge detector and histogram of oriented gradient method was a great combination with the ensemble classifier and the recognition rate. The experimental results had shown the robustness of our proposed model. Logistic Regression and Support Vector Machine have achieved 100% accuracy. The developed model was validated using two public datasets, and the findings have proved that our model outperformed other compared studies.

Keywords: Hand gesture recognition; canny edge detector; histogram of oriented gradient; ensemble classifier; majority voting

1 Introduction

Gesture can be described as a nonverbal communication using different parts of the body like hand and face. The definition of gesture is "the use of motions of the limbs or body as a means of expression; a movement usually of the body or limbs that expresses or emphasizes an idea, sentiment, or attitude" [1]. Posture, gestures, and body language are methods used when people need to interact with another who is deaf or speech-impaired [2,3].

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Nowadays, Hand Gesture Recognition (HGR) is widely used for human-computer interaction where gestures recognition of hand and face are used as a physical medium in such system. This system is based on the computer vision concept that other methods that do not rely on computer vision have also been developed like gloves and sensors but they are high-cost, complex, and difficult to be promoted [4]. There are many applications implement hand gesture recognition such as sign language recognition, robot control, vision-based virtual reality, interactive games, television control, etc. [5,6].

Hand gesture is divided into two types: static gesture (single posture or certain pose) that requires less complex computations and dynamic gesture (series of postures) that need more computations and it is more complex than static but suitable for real-time environments [4,7].

According to [8], the hand gesture recognition process can be summarized into three steps: First step is the image pre-processing and segmentation to removes unneeded data and divide the input image (hand gesture) into regions [9,10]. The next step is features extraction. Features are considered the essential components of any dataset, where every dataset includes a lot of features. Some of these features may be redundant or irrelevant to the problem that may have impact on the performance of the learning algorithms [11]. Different methods can be used to extract the features vector of the segmented image. Good image segmentation leads to a perfect features extraction, which plays a major role in gesture recognition success [12]. The last step is the gesture classification. Different classifiers can be used to recognize hand gestures among which: Hidden Markov Model (HMM) [13], support vector machine (SVM) [14], Artificial Neural Networks (ANN), Decision Tree (DT) [15], Logistic Regression (LR) [16], Naive Bayes (NB), and K Nearest Neighbor (KNN) [17] which can be trained to distinguish hand gestures.

Most of the hand gesture recognition systems that have been developed by many researchers last years used one or two machine learning techniques to recognize hand gestures with a reasonable accuracy and only a few studies proposed the ensemble learning approach. In addition to that, some of these systems were specifically designed for certain datasets and did not evaluate on other datasets, which means that the model may not succeed with other different hand gesture datasets.

This work suggests using the ensemble learning approach to build a powerful model that can be generalized to recognize several hand gestures with a 100% recognition rate. It proposes a robust way by combining the most powerful Machine Learning classification algorithms that are employed to recognize ten static hand gestures. The gestures represent the numbers from 1 to 5, and the up, down, right, left, and stop signs. Several classifiers are utilized to build the ensemble classification model. In combination, we have different types of classifiers, and different parameter values, which means that a variety of classification approaches are combined within an ensemble method. The results will be combined using the majority voting method to produce the final accuracy score for the model overall.

The structure of this paper is organized as follows: the next section takes a look over the previous hand gesture recognition research works. Section 3 provides the methodology of the proposed model. We explain the experimental results in Section 4 and discuss them in Section 5. Finally, the conclusion and future work suggestions are presented in the last section.

2 Related Works

In this section, we seek to cover the related works that have employed Machine Learning algorithms in different areas as well as the state-of-art studies that have developed in the field of hand gesture recognition.

2.1 *Machine Learning*

In recent years, Machine Learning algorithms have spread and became widely used in various domains. Many researchers have employed ML algorithms to predict or classify different problems. The authors in a novel study [18] developed a Machine Learning model to predict platform sales of the e-commerce system. They employed the Bidirectional Recurrent Neural Networks (BRNN) with a SFS-Guided Whale Optimization Algorithm (WOA) to optimize the weights of BRNN parameters. The results were compared using different optimization techniques such as Genetic Algorithm (GA), WOA, and Particle Swarm Optimization (PSO). While another study proposed a Machine Learning model for forecasting of wind speed by combining the Adaptive Dynamic Particle Swarm Algorithm (AD-PSO) with the Guided Whale Optimization Algorithm. The use of the AD-PSO-Guided WOA algorithm enhances the parameters of Long Short Term Memory (LSTM) classifier. Wilcoxon's rank-sum and ANOVA tests have employed to evaluate the result where the model achieved a high accuracy [19]. The authors in [20] proposed the Sine Cosine Dynamic Group (SCDG)-based voting classifier for operational risks identification in supply chain 4.0. They applied Support Vector Machine (SVM), Neural Network (NN), k-Nearest Neighbor (KNN), and Random Forest classifiers to identify the risks in the first experiment. Then, they compared the proposed voting SCDG method with the bagging and majority voting in the second experiment and with other optimization algorithms (PSO, WOA, GWO, and GA) in the last experiment to test the effectiveness of the method.

2.2 *Hand Gesture Recognition*

Several previous studies have been done on hand gesture recognition. Many researchers developed various techniques and methods that aimed to understand and recognize specific hand gestures.

Quan et al. developed in [8] a novel model for hand gesture recognition by employing Convolutional Neural Network (CNN). The image preprocessing is unnecessary with CNN. The researchers used a canny edge detection to remove the illumination from the hand gesture image. The edge of the hand gesture is detected after enhancing the image. To recognize the hand gesture effectively, the researchers developed a robust CNN architecture to describe the spatial hand features. Overall, the model contains 11 layers distributed as follows: one soft-max layer, 3 max-pooling layers, 2 full-connected layers, and 5 convolutional layers. The proposed method was effective and competitive and it achieved a 98.2% recognition rate.

Tiantian et al. [21] introduced a new method for hand gesture recognition by combining histograms of oriented gradients and the skin similarity. They found a new gradient by adding the skin similarity to the gradient of each pixel of the image to enhance the features. Hand gesture classification is done by applying the Support Vector Machine (SVM) classifier and employing different sizes of cells in the histograms of oriented gradients. The proposed method has improved the recognition rate and reached more than 91%.

In the latest years, the use of ensemble methods has quickly grown and imposed more attention from pattern recognition and different other domains according to the ability to maximize the accuracy of the prediction as well as the classification in the learning process. Schumacher et al. used ensemble classification methods for hand gesture recognition [22]. The researchers combined several classifier types along with different data feature sets such as the coordinates of the head relative to the positions of the hand, values of velocity, or path curves. Then the min and max values are used to transform the feature values into small values ranged between -1 and 1. For ensemble learning, the researchers employed a combination of three classifiers, which are: Support Vector Regression (SVR), the polynomial classifier (PC), Multi-Layer Perceptron (MLP) in order to classify hand gestures accurately. Then, they applied the Genetic search algorithm to identify the high effective classifier based on some optimization criteria. The performance of the proposed model showed that the use of the ensemble approach for gesture recognition was robust and effective to be used in different domains.

Mantecon et al. [23] built the leapGestRecog Dataset and used Support Vector Machine (SVM) for hand gesture recognition. They have employed the Depth Spatiograms of Quantized Patterns (DSQP) feature descriptor for images' feature extraction without any hand segmentation phase. Each gesture was trained by a single SVM classifier using positive examples of gestures that were included in the training data and negative samples of other gestures. Their proposed model achieved a 99% score.

Sharma et al. [24] proposed a simple deep learning approach using the Convolutional Neural Network (CNN) model. The model contains two Convolutional Layers, two Max-Pooling Layers, and two Dense/Fully connected Layers. The Convolutional Layers apply series of filters on raw images and perform a set of mathematical calculations to generate a feature map. The pooling layer is used to extract the image features in order to reduce the dimensions and decrease the execution time. Last, the dense layer is utilized to classify the hand gesture images. The proposed model achieved a high recognition rate of up to 99%.

Basha [25] used the Convolutional Neural Network method for hand gesture recognition and feature extraction in the LeapGestRecog dataset. The model consists of three main layers, one for each convolution layer, one max-pooling layer, and one fully connected layer. In the convolution layer, the author applied a Gaussian filter to extract the important features in addition to other different filters for edge detection and image blurring and sharpening. He used a stride of 2 and zero padding when applied the kernel filter to the input image. The max-pooling layer is very important to remove unwanted data by selecting the largest elements in the 2×2 matrix (as a stride value was 2) and replace it with this maximum value. Whereas the fully connected layer focuses on high-level features of the extracted features vector that correspond to a certain class. The accuracy obtained from the proposed model was 98%.

Butt et al. [26] developed a model that combined different machine learning algorithms that were applied using Weka and Rapid Miner software. The authors used HOG (Histogram of Oriented Gradient) and LBP (Local Binary Pattern) for extracting image features. They ran different experiments on the MNIST dataset to obtain the best results. The auto Rapid Miner model has achieved the best results with the following algorithms: K Nearest Neighbor (KNN), Naive Bayes (NB), Generalized Linear Model (GLM), and Deep Learning (DL). GLM was the highest with 100% accuracy.

Bilgin et al. [27] proposed a novel deep learning model using LeNet and CapsNet which are the state-of-art methods of convolutional neural network (CNN). The LeNet-5 model consists of 5 convolutional layers and 2 sub-sampling layers and the CapsNet is composed of one convolutional layer and 2 capsule layers. The authors carried out three experiments using the MNIST dataset in order to recognize sign language characters through training the two models on training data and evaluating them on the testing set. The performance of the CapsNet model has been improved and achieved 95% by augmented the training data with different parameters.

Sabeenian et al. [28] developed a custom CNN model to recognize sign language using the MNIST dataset. The construction of this model includes 11 layers, 4 convolutional layers, 3 max-pooling layers, 2 dense layers, one flatten layer, and the last one is a dropout layer. The images are first converted into grayscale and resized before they are fed to the trained model. The convolutional layers utilize (7×7) and (3×3) convolutional kernels. Whereas each max-pooling layer contains a (2×2) pool. The soft-Max layer classifies the images into different 24 classes and outputs the label as a text. The model produced 99% accuracy on the training data and more than 93% on the validation set.

In this study, we are using the ensemble learning approach by combining the most powerful classification algorithms for hand gesture recognition. Our experiments are using both datasets: LeapGestRecog and MNIST on top of our dataset and our results are compared with those provided in the previous models using the same datasets.

3 Proposed Model

The proposed model combines the most common Machine Learning classification algorithms to identify which gesture is performed by the hand of a given user.

The main contribution of this study includes:

- Create a complete model that can detect and recognize a hand gesture with reasonable accuracy.
- Build a simple system based on computer vision that does not need any special hardware.
- Recognize static hand gesture images that are captured with a simple and clear background and in stable illumination.

Fig. 1 displays the architecture of the developed model where the process involves five main stages to recognize the hand gesture:

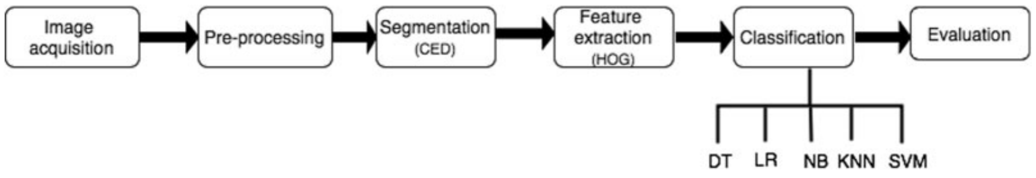

Figure 1: Proposed system architecture

3.1 Image Acquisition

Image Acquisition is the first stage in Digital Image Processing and Computer Vision Systems. In this work, the webcam of a laptop is used as a tool for acquiring the images. The size of all images is adjusted to be the same size as 500 × 600. Ten hand gestures are taken under different illumination with different backgrounds. The hand gestures that are recognized in this system represent the numbers 1, 2, 3, 4, and 5 and the up, down, right, left, and stop signs. Fig. 2 displays a sample of images that have been captured.

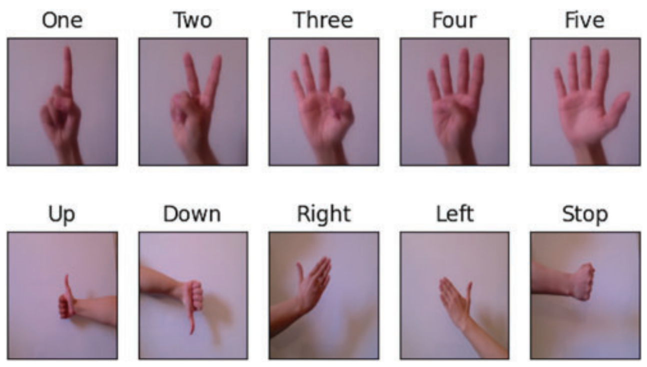

Figure 2: Original images of ten hand gestures

3.2 Image Preprocessing

Image pre-processing is fundamental for hand gestures recognition systems. It helps to remove irrelevant data, which leads to better and faster feature extraction. Additionally, it has positive effects on the results of image analysis and it also decreases the processing time. Image cropping is applied to the original pictures to remove the arm and focus on the palm only. Then the image size is reduced so all unneeded pixels are removed and lower data size is produced. The original image size is 500×600, which is resized to 150×200. The resulting image is then converted into grayscale to reduce the computations. Last, the noise is filtered using median filter.

3.3 Image Segmentation

Image segmentation means dividing the image into multiple disjoint regions, each region consists of a group of pixels that are similar in their characteristics such as color, intensity, or texture to effectively process and analyze the image that helps in object detection, classification, and recognition. Several techniques have been advanced which can be used for segmenting different types of images. All the proposed Image segmentation techniques so far were created for particular purposes. Therefore, there is no single segmentation approach that can be considered ideal for all image types. The choice of a proper technique can be defined based on the image type and the characteristics of the problem.

Canny Edge Detector is considered the most efficient and the oldest edge detection-based technique for image segmentation. It was first introduced by John F. Canny in 1986 and is based on the Gaussian operator that takes the second derivative for the image intensity function. The canny edge detector is widely used for edge detection and works effectively with images containing noise. This technique detects edges based on three basic criteria:

- Reduce the error probability to detect non-edge pixels or missing real-edge pixels.
- Good localization of edge points by making the space between the actual edge and the located position as little as possible.
- Only one single edge response should exist.

To apply Canny edge detector, five essential steps are needed:

- Image smoothing and noise removal using Gaussian filter.
- Find the image gradient.
- Apply non-maximum suppression to ensure that just the local upper limit represents the edges.
- Define the possible edges by applying a double threshold.
- Finalize the edge detection, which is tracking the edges by hysteresis and eliminate all weak and disjoint edges.

Fig. 3 shows the application of the Canny edge detector to our images.

3.4 Feature Extraction

Feature extraction (FE) can be defined as the process of transforming raw images into a set of useful features called features vector that represents the important information for analysis and classification. It is considered an essential process for any classification model, which aims to extract the relevant information that distinguishes each class from others.

Different FE techniques have been developed to detect the best features from original images and representing them in a simple representation without losing any important information.

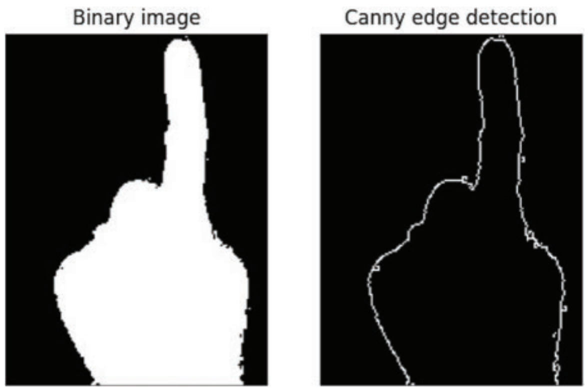

Figure 3: The application of canny edge detector

Histogram of Oriented Gradient (HOG) is the simplest and most efficient feature extraction technique. It was developed in 2005 by Dalal et al. [29]. HOG is a gradient-based feature descriptor that is broadly used for object recognition and detection. It describes the shape and appearance of an image and converts the image representation from the pixel-based into a gradient-based image. The implementation steps of the HOG descriptor can be described as follows:

- Split the given image into small parts like cells.
- Calculate a weighted vote for a histogram of gradient or orientations of edges for pixels that exist in each cell.
- Quantize every cell into angular bins depending on the direction of the gradient.
- Combine the adjacent cells into spatial blocks and normalizes each block histogram separately.
- Concatenate all those block histograms to represent the image feature descriptor.

The cell and block sizes differ based on detecting window size (image size). The ideal image size that was tested by Dalal and Triggs in the HOG original paper is 64×128 pixels, but any size that has a ratio of 1:2 can be dealt with.

In this work, the size of the images was adjusted to 128×256 pixels, and the cell size was chosen to be 8×8 with a 2×2 block size and 9 orientation bins. Fig. 4 shows the implementation of the HOG feature descriptor in our model.

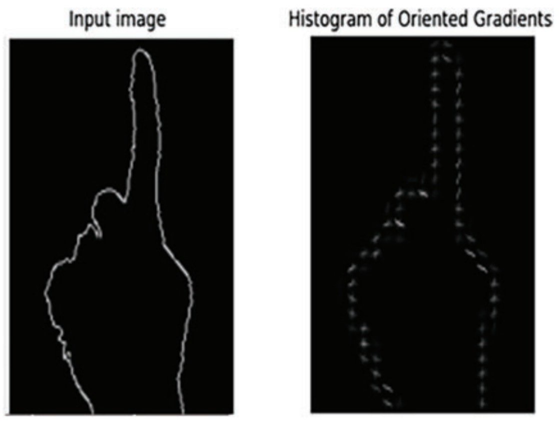

Figure 4: The HOG feature descriptor of the hand gesture image

Fig. 5 shows the workflow of image processing starting with pre-processing in the first row passing through image segmentation and then image feature extraction in the second row. The final output is the features vector of size 16740, which will be used later to feed the classification model.

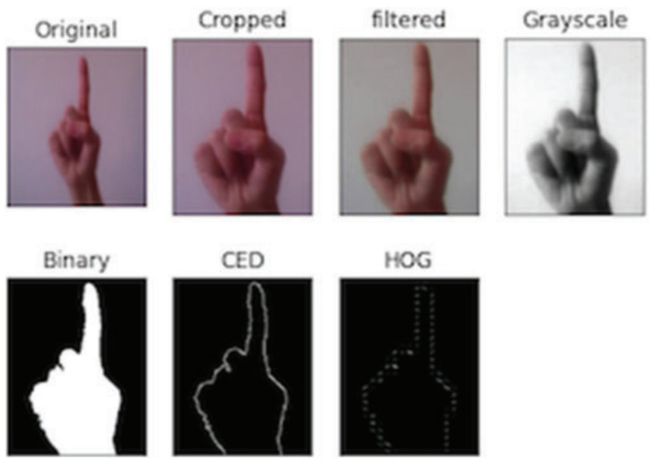

Figure 5: Workflow of image processing

3.5 Ensemble Classification

The ensemble learning method is a combination of multiple accurate and diverse machine-learning classifiers to build a robust classification model that outperforms the individual classifier performance. It aims to merge the predictions of multiple ML classifiers to enhance the performance of each classifier by building diverse classifiers individually and then calculate the average of their predictions using uniform averaging or voting methods [30]. Accuracy means the potential of each classifier to predict the class label that is as close as possible to the truth label. On the other hand, diversity can be created by employing a set of diverse classifiers with different learning capabilities. These two factors are considered when constructing an ensemble classifier [31]. The idea behind the ensemble method is that each classifier has a different learning ability and parameters values which make the accuracy of individual classifier differs from other classifiers. The ensemble model performance relies on the number of included classifiers and the accuracy generated by each individual classifier. The use of the ensemble approach helps to collect the best results for all produced accuracies and aggregate them using a simple or weighted majority voting method to generate the final classification result.

The ensemble classifiers used in this study consist of five state-of-arts base classifiers, which are: Decision Tree (DT), Logistic Regression (LR), Naive Bayes (NB), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). After which, we apply a majority voting ensemble method to find the final prediction result. Voting is the simplest method that combines predictions from various classifiers, which were trained and evaluated in parallel in order to get the advantages of each algorithm. In majority voting, the predictions for each class/label are summed and the class that achieved the largest number of votes is chosen as the final output.

3.6 Evaluation

Model Evaluation Metrics are used to quantify the performance of the constructed machine-learning model. Different metrics exist and the choice of one of them depends on the given algorithm task. The most common metrics used for classification are confusion matrix, accuracy, F-score, precision, recall, and Area Under Curve (AUC).

According to our model, we have separately measured each classifier performance and the overall ensemble model using four evaluation metrics (accuracy, F-score, precision, recall) that can be computed using Eqs. (1)–(4) respectively.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \tag{1}$$

$$\mathrm{F1} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{2}$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{3}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{4}$$

4 Experimental Results

Various experiments have been conducted on the constructed dataset in order to get the best results. The results are recorded and compared after each experiment.

4.1 Dataset Collection

In order to train the proposed model, 10 static hand gestures have been captured from 4 subjects for the left and right hands. Each gesture represents a class and each class contains 160 photos (10×160), thus the total images in the dataset are 1600. The gestures correspond to the numbers 1, 2, 3, 4, and 5, and the signs are left, right, up, down, and stop as were shown in Fig. 2. The images were taken with white and black backgrounds under sunlight and lamplight from different orientations. The dataset is split into training and testing sets to train and validate the model.

4.2 Results

The first experiment involves changing the amount of training and testing sets to select the best splitting criteria. The amount of training and testing sets affects the accuracy results and inappropriate splitting of data may cause an over-fitting problem. The first experimental results are recorded in Tab. 1.

In the second experiment, the parameters' values of each classifier are tweaked repeatedly for tuning their values that give the highest accuracy score. The selected values are used in the final model. The parameters that may affect classification results and their possible values are illustrated in Tab. 2.

The results obtained from the second experiment are summarized in Tab. 3. Each classifier is trained individually with its corresponding parameters and possible values. The values that achieve the highest accuracy score are chosen to use in the final model.

For the Decision tree classifier, the experiment was conducted into two phases where the criterion parameter takes the 'gini' value in the first phase and the 'entropy' value in the second phase. The value of max_depth parameter takes each time, one of the possible values mentioned in Tab. 2.

Table 1: The results of the first experiment

#	Splitting criteria	ML classifier			Majority voting
			Training	Testing	
1	Training set = 85% Testing set = 15%	DT	96.54%	94.58%	Training = 100.0% Testing = 100.0%
		LR	100.0%	100.0%	
		NB	97.28%	97.92%	
		SVM	100.0%	100.0%	
		KNN	99.12%	94.17%	
2	Training set = 80% Testing set = 20%	DT	94.69%	92.5%	Training = 100.0% Testing = 100.0%
		LR	100.0%	100.0%	
		NB	97.42%	96.56%	
		SVM	100.0%	100.0%	
		KNN	97.66%	88.75%	
3	Training set = 75% Testing set = 25%	DT	98.5%	95.0%	Training = 100.0% Testing = 100.0%
		LR	100.0%	100.0%	
		NB	97.25%	95.75%	
		SVM	100.0%	100.0%	
		KNN	97.92%	91.75%	
4	Training set = 70% Testing set = 30%	DT	97.05%	94.38%	Training = 100.0% Testing = 100.0%
		LR	100.0%	100.0%	
		NB	97.41%	95.62%	
		SVM	100.0%	100.0%	
		KNN	95.27%	86.88%	
5	Training set = 65% Testing set = 35%	DT	97.02%	93.04%	Training = 100.0% Testing = 99.29%
		LR	100.0%	100.0%	
		NB	97.4%	94.11%	
		SVM	100.0%	100.0%	
		KNN	96.06%	89.11%	

The experiment of the Logistic Regression classifier was also carried out in two phases. First, the value of solver is tweaked. In the second stage, the value of the solver is set to 'newton-cg' and C takes one of the predetermined values in Tab. 2.

In the Naive Bayes experiment, alpha takes one of the values in the range defined in Tab. 2 and the fit_prior value is changed between true and false.

Table 2: ML classifiers' parameters with their possible values

ML classifier	Parameter	Possible value
DT	criterion	{'gini' , 'entropy'}
	max_depth	{2, 5, 10, 15, 20}
LR	Solver	{['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}
	C	{1, 5, 10, 15, 20}
NB	Alpha	{0.00001, 0.0001, 0.001, 0.01, 0.1, 1}
	Fit_prior	{True, False}
SVM	kernel	{'rbf', 'sigmoid' ,'poly'}
	C	{0.001, 0.01, 0.1, 1.0, 10, 100}
	gamma	{0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 'scale', 'auto'}
KNN	n_neighbors	{3, 5, 10, 15, 20, 25, 30}

Table 3: The results of the second experiment

ML classifier	#Experiment	Accuracy score
DT	1	[0.234375, 0.54375, 0.921875, 1.0, 1.0]
	2	[0.26875, 0.728125, 0.9875, 0.9875, 0.9875]
LR	1 and 2	[1.0, 1.0, 1.0, 1.0, 1.0]
NB	1 and 2	[0.9875, 0.984375, 0.98125, 0.971875, 0.9625, 0.93125]
SVM	1, kernel = 'rbf'	[0.075, 0.075, 0.7625, 1.0, 1.0, 1.0]
	2, kernel = 'sigmoid'	[0.075, 0.075, 0.86875, 1.0, 1.0, 1.0]
	3, kernel = 'poly'	[0.075, 0.075, 0.50625, 0.9875, 0.9875, 0.9875]
	4, C = 1	[1.0, 1.0, 0.9875, 0.9875, 0.95, 0.8625, 0.85625, 0.85625]
	5, C = 10	[1.0, 1.0, 0.9875, 0.9875, 0.95, 0.8625, 0.85625, 0.85625]
	6, C = 100	[1.0, 1.0, 0.9875, 0.9875, 0.95, 0.8625, 0.85625, 0.85625]
KNN	1	[0.990625, 0.909375, 0.875, 0.85, 0.83125, 0.821875, 0.778125]

For the Support Vector Machine, we did six experiments that are divided into two parts. In the first part, we focused on changing the values of kernel and C according to the predefined values. In the second part, we set the value of kernel to 'rbf', C takes one of these best values [1, 10, 100], and we changed the value of gamma to be one of the list ['scale', 'auto', 0.001, 0.01, 0.1, 1.0, 10.0, 100.0].

According to the K-Nearest Neighbors, the only parameter that is selected to do the experiments is n_neighbors, which indicates the number of neighbors that will use for Kneighbors( ). The experiment involves changing the value of n_neighbors. The smallest values of n_neighbors give the best performance of KNN, whereas the accuracy is sharply decreased when its value becomes high.

The best parameters' values for each classifier that have been obtained from the previous experiments were picked to carry out the third experiment. After tuning the values of each classifier's parameters, the five classifiers are combined to build the final model. The dataset in this experiment is split into 80% (training set) and 20% (testing set). The training set includes 1280 images with 16740 (128×256) features. The testing set contains 320 images with 16740 features. The findings are recorded in Tab. 4.

Table 4: The results of the third and fourth experiments

ML classifier	Selected parameter	Accuracy score	Precision	Recall	F1-score	The mean score (10-fold CV)
Decision tree	Criterion = 'gini' max_depth = 20	99.38%	99.0%	99.0%	99.0%	0.98 (+/− 0.12)
Logistic regression	Solver = 'newton-cg'	100.0%	100.0%	100.0%	100.0%	1.00 (+/− 0.00)
Naïve bayes	Alpha = 0.00001 Fit_prior = True	97.19%	98.0%	98.0%	98.0%	0.95 (+/− 0.07)
K-Nearest neighbors	n_neighbors = 3	100.0%	100.0%	100.0%	100.0%	0.87 (+/− 0.09)
Support vector machine	Kernel = 'rbf' C = 10 Gama = 'scale'	100.0%	100.0%	100.0%	100.0%	1.00 (+/− 0.02)
Majority voting	Voting = 'soft'	100.0%	100.0%	100.0%	100.0%	1.00 (+/− 0.02)

The performance of LR, SVM, and KNN outperforms the DT and NB classifiers in the final model. The Naive Bayes was the weakest among the five classifiers. The overall accuracy of the ensemble classification model is computed by applying the majority-voting algorithm. The ensemble method can improve results and provide more accurate solutions than that a single model produced. The majority voting is one of the easiest ensemble methods used for classification. The five employed classifiers represent the inputs of the voting method. Then, each classifier makes a prediction called votes for the test samples. The prediction that achieves higher than half of the votes is considered as the final output of the ensemble model.

In the fourth experiment, we used the K-fold cross-validation function to partition the training set into K smaller sets. The model is training on K-1 sets and testing on the remaining set. Then, the performance is measured by computing the average of values computed at each iteration. The value of K in this experiment is set to 10 (10-fold CV). The obtained results were listed in Tab. 4. The accuracy of LR and SVM remains the same (100.0%) and they keep the high performance. The KNN accuracy was 87.0%, it sharply decreased and there is a big difference compared to the previous experiment. The difference between DT and NB accuracies in this experiment and the previous one does not exceed two degrees.

4.3 Model Evaluation

In order to assess our proposed model and contrast the results of this study with other researchers' works, two publicly available datasets were used.

4.3.1 leapGestRecog Dataset

The first dataset is called ‘leapGestRecog’. It was created by Mantecon et al. [23]. The dataset consists of 10 different hand gestures that were captured from 10 individuals (5 men and 5 women). The Leap Motion sensor was employed to acquire the images. The number of images in the presented dataset is equal to 20,000. A number of 200 images were taken per each gesture and subject. Fig. 6 shows a sample of each hand gesture included in this dataset. In the figure, the hand gestures from left to right refer to Palm, Fist, Fist Moved, Thumb, Index, OK, Palm_moved, Down, C letter, and I letter.

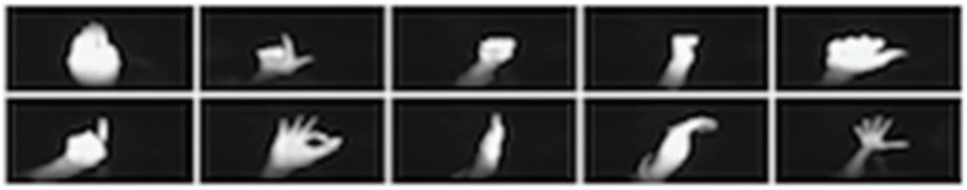

Figure 6: Samples of leapGestRecog dataset [23]

We have applied our proposed model to this dataset and compared our results with other methods that have been discussed in the related work section. We compared the results using four evaluation metrics (accuracy, F-score, precision, recall). Tab. 5 summarizes the works that have been applied to LeapGestRecog dataset.

Table 5: The description of the proposed method and others’ methods on leapGestRecog dataset

Study	Segmentation method	Feature extraction method	Classification method
Mantecon et al. [23]	- - - - - - -	Depth spatiograms of quantized patterns (DSQP) feature descriptor	SVM
Sharma et al. [24]	Adaptive skin-color model switching method (ASSM).	Determine the hand-pose angles using the Gabor filter.	CNN
Basha [25]	- - - - - -	CNN	CNN
Proposed method	Canny edge detection.	Histogram of oriented Gradient (HOG).	Ensemble method (DT, LR, NB, KNN, and SVM)

The results of the proposed model compared to other methods that have been applied to the leapGestRecog dataset are shown in Tab. 6.

Table 6: Results of the proposed method and others' methods on leapGestRecog dataset

Study	Method	Accuracy	F-score	Precision	Recall
Mantecon et al. [23]	SVM	99.0%	99%	99%	99%
Sharma et al. [24]	CNN	99.95%	99.0%	98.0%	99.0%%
Basha [25]	CNN	98.0%	...	...	...
Proposed ensemble method	DT	100.0%	100.0%	100.0%	100.0%
	LR	100.0%	100.0%	100.0%	100.0%
	NB	98.44%	98.0%	99.0%	98.0%
	KNN	97.81%	98.0%	98.0%	98.0%
	SVM	100.0%	100.0%	100.0%	100.0%
	Voting	100.0%	100.0%	100.0%	100.0%

As shown in Tab. 6, the results of our proposed model achieved a 100% in most of the used classifiers except the NB and KNN. The results of [23,24] were close together, while they are higher than the [25] work. The ensemble approach that is proposed in our model outperforms other compared methods. In addition, the use of the canny edge detector for image segmentation and a Histogram of Oriented Gradient (HOG) for feature extraction improved the recognition rate for the leapGestRecog dataset.

4.3.2 Sign Language MNIST Dataset

The second dataset that is used to evaluate our model is called 'Sign Language MNIST' that is available online on the well-Known source "Kaggle" [32]. The dataset represents the hand gestures of the American Sign Language letters with 24 classes (except J and Z letters as they require motion). It consists of 9 folders, which refer to the subjects and each folder contains 240 color images and one csv file. Fig. 7 shows a sample of all signs with their corresponding labels. The raw images in this dataset need to be processed (cropped, resized, and convert into grayscale and binary images) before using them in our model.

We have applied our proposed model to the MNIST dataset and compared our results with other recent methods that have been mentioned in the related work section. We also here used the same four evaluation metrics (accuracy, F-score, precision, recall) to compare the results. Tab. 7 shows a short description of the works that have been applied to the MNIST dataset.

Tab. 8 gives the comparison between the results of our model and the state-of-art models that have been applied to recognize hand gestures of American Sign Language using the MNIST dataset.

Our proposed model obtained the highest accuracies for the most used classifiers. Logistic Regression, Support Vector Machine, and K Nearest Neighbors have achieved a 100% rate. The proposed model outperforms other state-of-arts methods that were applied to MNIST dataset in the presented studies.

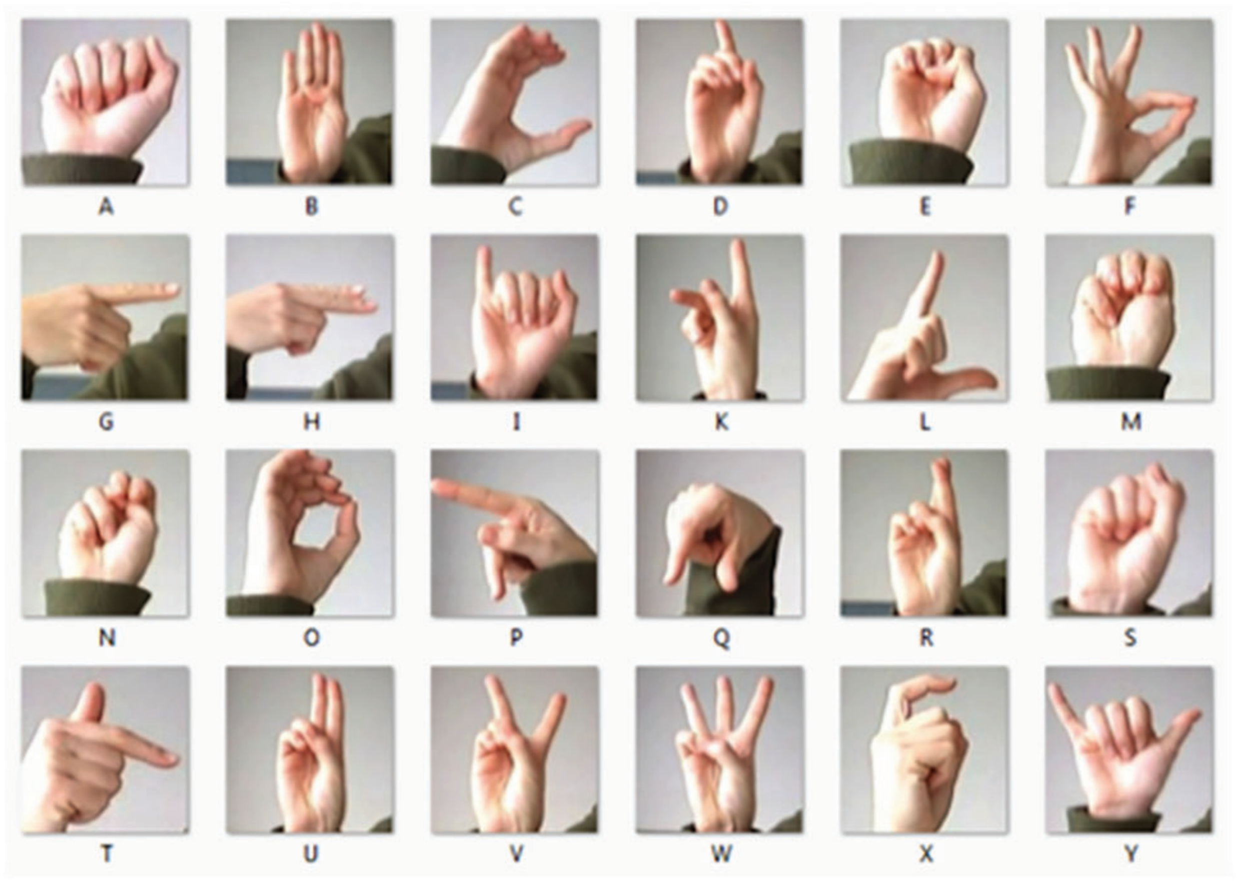

Figure 7: Sign language MNIST dataset samples [26]

Table 7: The description of the proposed model and other state-of-art models applied to MNIST dataset

Study	Segmentation method	Feature extraction method	Classification method
Butt et al. [26]	- - - - - - -	Histogram of oriented gradient -Local binary pattern	-Weka (NB, Lazy IBK, and RF) -Rapid miner (KNN, Neural Net, generalized leaner model, deep learning, NB, RF, and DT)
Bilgin et al. [27]	- - - - - - -	- - - - - - -	LeNet and CapsNet
Sabeenian et al. [28]	Thresholding.	- - - - - -	CNN
Proposed method	CED.	HOG	Ensemble method (DT, LR, NB, KNN, and SVM)

Table 8: The results of the proposed model and other state-of-art models applied to the MNIST dataset

Study	Classification method	Accuracy	F-score	Precision	Recall
Butt et al. [26]	KNN	98.03%	. . . .	. . . .	. . .
	NB	89.5%	. . .	. . .	. . .
	GLM	100.0%	. . .	. . .	. . .
	DL	99.9%	. . .	. . .	. . .
Bilgin et al. [27]	LeNet	82.19%	80.95	81.24%	81.82%
	CapsNet	88.93%	86.41%	84.48%	89.04%
	CapsNet augmented	95.08%	93.22%	91.11%	95.63%
Sabeenian et al. [28]	CNN	93.0%	. . .	. . .	. . .
Proposed method	DT	80.95%	86.0%	97.0%	81.0%
	LR	100.0%	100.0%	100.0%	100.0%
	NB	99.11%	99.0%	99.0%	99.0%
	KNN	98.81%	98.81%	99.0%	99.0%
	SVM	100.0%	100.0%	100.0%	100.0%
	Voting	100.0%	100.0%	100.0%	100.0%

5 Discussion

According to the hand gestures recognition systems, the major task is determining and extracting the real edges successfully. The edge detection process helps to represent the hand details as a single unit, which makes the image analysis and feature extraction easier. For that reason, we decided to use the edge-detection-based image segmentation technique in this study. Canny edge detector is the best choice for the majority of object recognition applications. It can extract edges from noisy images and detect even weak edges effectively by applying the Gaussian filter.

For the features extraction, the Histogram of Oriented Gradient (HOG) was used to form feature vectors. The feature vectors help to recognize images effectively and have a big influence on the accuracy of the recognition.

There is no doubt that the use of the ensemble method and combining different classification models has a great impact on the improvement of recognition accuracy as we saw in the performed experiments. The application of a single classifier individually may not give the desired results. From all the experiments that have been conducted and the results that were acquired we can conclude some points:

In the first experiment, we investigate that the way of splitting the dataset into training and testing sets has a big impact on the accuracy scores and it may cause an overfitting problem with some classifiers. The best splitting for our proposed model was 85% for training and 15% for testing.

The values of the parameters must be tuned carefully for each classifier because they affect the classification results. As we see in the DT model, the value of criterion is important, and the accuracy increases obviously by increasing the value of max_depth until it reaches the peak (100%) when its value is equal to 15 or 20. Whereas LR results did not affect by changing the parameters, and it remained stable at a 100% rate. On the other hand, there is a steepest descent in the results of the NB model when the value of alpha was equal to 1, which means that the smallest alpha value gives the highest accuracy score. Three parameters have a significant impact on the SVM results: 1) kernel, 2) C, and 3) Gamma. There is a slight difference between 'rbf' and 'sigmoid' results and the value of C must be greater than or equal to 1 to achieve a 100% score with these two kernel values. The 'scale' and 'auto' values of gamma are the best, and we observed that the accuracy is reduced when we set a float value to gamma. The number of neighbors in the KNN model must be small and as the value increases the accuracy rate decreases.

In the last two experiments, we noticed a significant improvement in the accuracy scores of Decision Tree, Naive Bayes, and K-nearest Neighbors classifiers after tuning their parameters to the best values that we extracted from the previous experiment. On the other hand, we can see that the use of K-fold cross-validation slightly decreased the accuracy of DT and NB whereas the KNN model achieved a lower score than other scenarios when using k-fold CV.

6 Conclusion and Future Work

Hand gesture recognition is the process of recognizing and interpreting different hand gestures of the given input data into meaningful expressions and identifying which gesture is performed by a given user. In this work, we developed a robust model to recognize hand gestures using a combination of Machine Learning classification algorithms with the highest recognition rate. The use of canny's edge detector for image segmentation and Histogram of Oriented Gradient (HOG) for feature extraction improved the recognition rate. The experimental results had shown good recognition rates of Logistic Regression and Support Vector Machine with 100% accuracy in all the conducted experiments, which outperformed other classifiers. Additionally, our model was evaluated using two public datasets, where the performance of our model exceeded the results of other studies in both two datasets.

For future work, we suggest building a real-time hand gesture recognition model and utilizing various image segmentation techniques. The model can detect the hand even in a noisy background. The selection of the appropriate technique will be according to the image type.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. P. Sharma and G. K. Verma, "Human computer interaction using hand gesture," *Procedia Computer Science*, vol. 54, no. 4, pp. 721–727, 2015.

[2] R. Z. Khan and N. A. Ibraheem, "Hand gesture recognition: A literature review," *International Journal of Artificial Intelligence & Applications*, vol. 3, no. 4, pp. 161–174, 2012.

[3] A. S. Al-Shamayleh, R. A. Ahmad, M. A. M. Abushariah, K. A. Alam and N. Jomhari, "A systematic literature review on vision based gesture recognition techniques," *Multimed Tools Appl.*, vol. 77, no. 21, pp. 28121–28184, 2018.

[4] J. Shukla and A. Dwivedi, "A method for hand gesture recognition," in *Proc. 2014 Fourth Int. Conf. on Communication Systems and Network Technologies CSNT 2014*, Bhopal, India, pp. 919–923, 2014.

[5] D. L. Dinh, H. J. Jeon, S. B. Nam, S. Lee and T. S. Kim, "Hand number gesture recognition using the recognized hand parts in depth images," *Multimed Tools Appl.*, vol. 75, no. 2, pp. 1333–1348, 2016.

[6] R. F. Rahmat, T. Chairunnisa, D. Gunawan, M. F. Basha and R. Budiarto, “Hand gestures recognition with improved skin color segmentation in human-computer interaction applications,” *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 3, pp. 727–739, 2019.
[7] C. Quan and J. Liang, “A simple and effective method for hand gesture recognition,” in *Proc. 2016 6th Int. Conf. on Digital Home, ICDH 2016*, Wuhan, China, pp. 302–305, 2017.
[8] C. Quan and J. Liang, “A robust hand gesture recognition method via convolutional neural network,” in *Proc. 2016 Int. Conf. on Network and Information Systems for Computers ICNISC 2016*, Guangzhou, China, pp. 64–67, 2017.
[9] M. Sonka, V. Hlavac and R. D. Boyle, “Image pre-processing,” in *Image Processing, Analysis and Machine Vision*, 4th ed. Stamford, USA: Cengage, Ch. 5, pp. 116–120, 2008. [Online]. Available: http://www.cengage.ca/c/isbn/9781305498686/.
[10] S. Saini and K. Arora, “A study analysis on the different image segmentation techniques,” *International Journal of Information & Computation Technology*, vol. 4, no. 14, pp. 1445–1452, 2014.
[11] M. M. Eid, E. M. El-kenawy and A. Ibrahim, “A binary sine cosine-modified whale optimization algorithm for feature selection,” in *Proc. 2021 National Computing Colleges Conf. (NCCC)*, Taif, Saudi Arabia, pp. 1–6, 2021.
[12] A. O. Salau and S. Jain, “Feature extraction: A survey of the types, techniques, applications,” in *Proc. 2019 Int. Conf. on Signal Processing and Communication ICSC 2019*, Noida, India, pp. 158–164, 2019.
[13] L. E. Baum and T. Petrie, “Statistical inference for probabilistic functions of finite state markov chains,” *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.
[14] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 2, pp. 273–297, 1995.
[15] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Boca Raton, USA: Routledge, 1984. [Online]. Available: https://doi.org/10.1201/9781315139470.
[16] J. S. Cramer, “The origins of logistic regression,” *Tinbergen Institute Discussion Paper*, vol. 119, no. 4, pp. 111–116, 2002.
[17] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
[18] A. A. Salamai, A. A. Ageeli and E. M. El-kenawy, “Forecasting E-commerce adoption based on bidirectional recurrent neural networks,” *Computers, Materials & Continua*, vol. 70, no. 3, pp. 5091–5106, 2022.
[19] A. Ibrahim, S. Mirjalili, M. El-said, S. S. Ghoneim, T. F. Ibrahim *et al.,* “Wind speed ensemble forecasting based on deep learning using adaptive dynamic optimization algorithm,” *IEEE Access*, vol. 9, pp. 125787–125804, 2021.
[20] A. A. Salamai, E. M. El-kenawy and I. Abdelhameed, “Dynamic voting classifier for risk identification in supply chain 4.0,” *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3749–3766, 2021.
[21] L. Tiantian, S. Jinyuan, L. Runjie and G. Yingying, “Hand gesture recognition based on improved histograms of oriented gradients,” in *Proc. 2015 27th Chinese Control and Decision Conf. (CCDC)*, Qingdao, China, pp. 4211–4215, 2015.
[22] J. Schumacher, D. Sakic, A. Grumpe, G. A. Fink and C. Wohler, “Active learning of ensemble classifiers for gesture recognition,” in *Lecture Notes in Computer Science*, vol. 7476. A. Pinz, T. Pock, H. Bischof, F. Leberl (Eds.), Berlin, Heidelberg: Springer, pp. 498–507, 2012.
[23] T. Mantecón, C. R. del-Blanco, F. Jaureguizar and N. García, “Hand gesture recognition using infrared imagery provided by leap motion controller,” *Advanced Concepts for Intelligent Vision Systems*, vol. 10016 LNCS, pp. 47–57, 2016.
[24] H. K. Sharma, P. Kumar, P. Ahlawat, N. Parashar, R. Soni *et al.,* “Deep learning based accurate hand gesture recognition using enhanced CNN,” *Journal of Critical Reviews*, vol. 8, no. 8, pp. 676–678, 2021.
[25] M. B. Basha, “Hand poses detection using convolutional neural network,” *International Journal of Scientific & Technology Research*, vol. 9, no. 1, pp. 1887–1891, 2020.
[26] U. M. Butt, B. Husnain, U. Ahmed, A. Tariq, I. Tariq *et al.,* “Feature based algorithmic analysis on American sign language dataset,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, pp. 583–589, 2019.

[27] M. Bilgin and K. Mutludogan, “American sign language character recognition with capsule networks,” in *Proc. 2019 3rd Int. Symp. on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Ankara, Turkey, pp. 1–6, 2019.

[28] R. S. Sabeenian, S. S. Bharathwaj and M. M. Aadhil, “Sign language recognition using deep learning and computer vision,” *Journal of Advanced Research in Dynamical & Control Systems*, vol. 12, no. 5, pp. 964–968, 2020.

[29] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, CVPR 2005*, San Diego, CA, USA, pp. 886–893, 2005.

[30] S. S. Ghoneim, T. A. Farrag, A. A. Rashed, E. M. El-kenawy and A. Ibrahim, “Adaptive dynamic meta-heuristics for feature selection and classification in diagnostic accuracy of transformer faults,” *IEEE Access*, vol. 9, pp. 78324–78340, 2021.

[31] M. Z. Jan and B. Verma, “A novel diversity measure and classifier selection approach for generating ensemble classifiers,” *IEEE Access*, vol. 7, pp. 156360–156373, 2019.

[32] Tecperson, (2017 Oct. 20). “Sign Language MNIST.” *Kaggle*. [Online]. Available: https://www.kaggle.com/datamunge/sign-language-mnist. [Accessed: 30-Dec-2020].