The Performance of Individual and Ensemble Classifiers for an Arabic Sign Language Recognition System

Miada A. Almasre
Department of Computer Science
Faculty of Computing and Information Technology
King AbdulAziz University
Jeddah, Saudi Arabia

Hana Al-Nuaim
Department of Computer Science
Faculty of Computing and Information Technology
King AbdulAziz University
Jeddah, Saudi Arabia

***Abstract*—The objective of this paper is to compare different classifiers' recognition accuracy for the 28 Arabic alphabet letters gestured by participants as Sign Language and captured by two depth sensors. The accuracy results of three individual classifiers: (1) the support vector machine (SVM), (2) random forest (RF), and (3) nearest neighbour (kNN), using the original gestured dataset were compared with the accuracy results using an ensemble of the results of each classifier, as recommended by the literature. SVM produced higher overall accuracy when running as an individual classifier regardless of the number of observations for each letter. However, for letters with fewer than 65 observations each, which created a far smaller dataset, RF had higher accuracy than SVM did when using the ensemble approach. Although RF produced higher accuracy results for classes with limited class observation data, the difference between the accuracy results of RF in phase 2 and SVM in phase 1 was negligible. The researchers conclude that such a difference does not warrant using the ensemble approach for this experiment, which adds more processing complexity without a significant increase in accuracy.**

Keywords*—Ensemble; Stacking; Support vector machine; SVM; Random forest; RF; Nearest neighbour; kNN; ArSL recognition system; Depth sensors

I. Introduction

Researchers in the Arab world, as well as researchers worldwide, are always investigating the use of assistive communication tools that could help the hearing-impaired in their daily lives when using their local languages and dialects. Although research has been done on using sign language recognition systems, limited research has addressed gesture recognition of Arabic Sign Language (ArSL). Also, few attempts have been made to develop a recognition system that can use a machine learning approach to interpreting ArSL letters [1].

Machine learning is "an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment" [2] [3]. Machine learning is more than just calculating averages or performing data manipulation; it involves creating predictions about observations based on previous information [3]. Using machine learning in gesture recognition involves four steps: 1) choosing appropriate sensors for collecting the gestured letters; 2) analysing and extracting features from the data, which are values related to describing the gestured letters; 3) classifying the data by recognizing and interpreting the gestures using one or multiple algorithms; and 4) displaying the recognised gesture's name by text or audio [4].

Also, machine learning can use either supervised or unsupervised learning to transfer sign language gestures into text format [5]. The supervised learning term refers to the fact that the algorithm was fed by a dataset in which the correct answers were given; then, the dataset was divided into two subsets: a "training dataset," which is used to build predictive models, and a "testing dataset," which is used to assess the performance of the model in the training step [6]. On the other hand, in unsupervised learning, the machine is not provided with knowledge about the model. The implemented algorithms classify the data to any instantaneous incoming hand or finger features [5].

In classifying segments, the observed gestured letters are placed into different classes based on the same or related values [7]. The collected data are divided into two sets: training and a testing set [7]. Therefore, classification is the process of assigning a new gestured letter to a specific class on the basis of training set values.

Many classifier algorithms exist, such as the neural network, support vector machine (SVM), nearest neighbour (kNN), and random forest (RF). Each has a different method for predicting or choosing the set to which a particular observation belongs [5].

Classifying data in machine learning can use either raw data with one algorithm or a combination of the results (predictions) of multiple algorithms, called "an ensemble," which is fed into an algorithm. Different ensemble models are available, with the most popular being: majority voting, bagging, boosting, and stacking [5].

Majority voting, considered the simplest, is a decision rule that chooses alternatives that have popular or majority votes [8] [9]. Bagging is a method of decreasing the variance of a prediction, boosting is a method of decreasing the bias of a predictive model and improving the predictive force, and stacking is similar to boosting by applying several models on the original data [9]. However, stacking takes the final

prediction using functions such as the sum, the average, or the weight of the predictions that other algorithms have generated [10].

Different types of stacking exist: some types use the original data together with classifiers as input to an ensemble model, whereas some do not. In addition, some use hard labels from classifiers, whereas others use probabilities [10]. Although using the ensemble approach requires mathematical complexity, it may increase the accuracy of the recognition. To classify gestures, one can use an individual classifier or an ensemble of the output of multiple classifiers. Results within the literature of classification using multiple learning algorithms or an ensemble model usually had higher accuracy rates, yielding better predictive performance than those obtained from the other formed learning algorithms [5].

Despite the complexity, the possible reasons for using an ensemble approach are: the data volume is too large or small, or not enough data are available to divide and conquer the data or for data fusion [11]. Therefore, in some cases, if the data to be analysed are too large, the use of one classifier may not effectively process the data. Similarly, ensemble systems can be used to address the exact opposite problem of not having enough data [12].

Analogically speaking, creating an additional step by feeding a classifier an ensemble of the data is like seeking a second and third opinion when it comes to a medical consultation: it increases reliability and reduces the risk of a wrong diagnosis [12].

The research methodology of Al-Masre and Al-Nuaim for gesture recognition used only one classifier (SVM) as a supervised machine learning hand-gesturing model [13] to classify the 28 letters (considered classes) of the Arabic alphabet “Figure 1.” In addition, to overcome the time complexity of interpreting the data for their model, the researchers used the principle component analysis (PCA) algorithm to simplify the large dataset by reducing features. Recognition results were at 86% for the ArSL letters tested in their experiment [13].

Although this research also used SVM to classify the 28 ArSL letters as in Al-Masre and Al-Nuaim [13], and to overcome the limitation of using the PCA algorithm, the proposed model focused on including all of the features of the collected data while adding a classification step, as recommended by the literature, to produce higher recognition accuracy. The extra step used the same classifiers that used the original dataset to classify the combined results (ensemble).

Therefore, it is the objective of this research to compare the recognition accuracy of three different popular individual classifiers using the original gestured dataset with the accuracy results of the same three classifiers using an ensemble of the results of the same classifiers.

In an attempt to investigate if adding a classification step produces higher accuracy, this research combined the results from three individual classifiers that used raw gestured data. The extra step would classify the combined (ensemble) data using the same three classifiers that used the original data.

The rest of the paper is organised as: Section 2 and 3 present the literature surveying the overview of relevant work and the three classification algorithms used. Section 4 presents the research design and methodology used to complete the experiment. Finally, Section 5 discusses the results and presents the conclusion.

Fig. 1. the 28 Arabic Sign Language Alphabet

II. Literature Review

Many researchers have investigated the combination of voting schema since 1998, such as Kearns and Valiant [14], Rob Schapire, and others [15]. Schapire (1999) came up with an algorithm to apply such a combination called boosting, which is used with machine learning [15].

Ensemble learning has attracted considerable attention due to its good generalisation performance. The main issues in constructing a powerful ensemble include training a set of diverse and accurate base classifiers outputs and effectively combining them [12].

Ensemble majority vote, computed as the difference between the vote numbers that the correct class received and those of another class that received the most votes, is widely used to explain the success of ensemble learning. This definition of the ensemble margin does not consider the classification confidence of base classifiers [12].

Other ensemble algorithms appeared within the literature and were used in the machine learning field, such as boosting, AdaBoost, bagging, a mixture of experts, and stacked generalisation [16].

Using the stacking method, one can train a learning algorithm to combine the predictions of other learning algorithms. Firstly, all of the used algorithms are trained using the original data. Then, one makes a final prediction using all the predictions of the other algorithms (re-sampling) as inputs. The re-sampling method can be one of the following: sum, maximum, minimum, and weighted majority voting of the predictions that the other algorithms have generated as extra inputs [17].

The basis of ensemble methodology is simply creating a predictive model by integrating multiple models. It can be used to improve prediction performance; for example, researchers

from various disciplines, such as statistics, computer vision, and artificial intelligence, can use it [12].

Li, Hu, Wu, and Yu (2014) explored the influence of the classification confidence of the base classifiers in ensemble learning and had some interesting conclusions. First, they extended the definition of an ensemble margin based on the classification confidence of the base classifiers. Then, an optimisation objective was designed to compute the weights of the base classifiers by minimizing the margin-induced classification loss. They attempted several strategies to use the classification confidences and the weights. They observed that weighted voting based on classification confidence is better than simple voting if all of the base classifiers are used [17].

Farooq and Sazonov (2016) studied the ensemble performance of three classifiers—logistic regression, linear discriminant analysis, and decision trees—using three different ensemble approach: (1) boosting, (2) stacking, and (3) bagging. According to their results, the ensemble performance was enhanced by 4% compared to the individual algorithms [18].

In addition, Woźniak, Graña, and Corchado (2014) presented the idea of creating a multiple classifier system (MCS). They stated that no single classifier modelling approach that is optimal for all pattern-recognition tasks exists. Thus, MCS exploits the strengths of the different classifier models to create a high-quality compound recognition system, thus overcoming the performance of separate classifiers [19].

Ensembling is also known under various other names, such as multiple classifier systems, a mixture of experts, or a committee of classifiers [11]. Ensemble systems have shown to have higher performance in many applications compared to a single classifier's performance [11].

Most of the ensemble methods use a special mathematical model. Moreover, in applying the stacking method, researchers can use different types or scenarios—for example, combining the results of classifiers as a class label name, combining them as class prediction values, or combining the original dataset with class prediction values [20].

III. Classification Algorithms

A. Support Vector Machine (SVM)

The SVM algorithm is used to classify data by drawing a clear line between observation data, which are actually points on a plane. The margin space around the line should be as wide as possible to avoid the misclassified values of a testing set [21]. In addition, the SVMs can efficiently perform non-linear classification using what is called the kernel function, implicitly mapping its inputs into high-dimensional feature spaces [22].

Predicting the values and setting the kernel function parameters with correct values are the main objective of the SVM learning algorithm. Many statistical packages establish those parameters to give the best prediction, such as the R studio statistical package [23].

Using SVM requires choosing the parameter C (cost function) or a penalty term. It is used because SVM relies on predictions to make a decision about the best boundary that could cause an error. If the value of C is very large, then the decision boundary will be close to the data points nearest the support vectors. That means the misclassification probability increases as the value of C decreases [23].

B. k Nearest Neighbor (kNN)

The Nearest Neighbour (NN) algorithm for learning has worked on numeric feature values. NN treats values as distance metrics and uses them as standard definitions between instances [24]. A k-Nearest Neighbours algorithm (kNN) is a non-parametric method used for classification where the input consists of the k closest training examples in the feature space [25]. As a classifier, kNN allocates a pattern to the class of the nearest pattern value [26]. It starts with every observation in the training set as a prototype and then successively merges any two nearest patterns of the same class as long as the recognition rate is not reduced [27].

C. Random Forest (RF)

The term "random forest" refers to a collection of many decision trees (forest) where, when building at each node, there is some randomness in selecting the attribute to split. Thus, the RF breaks down a dataset into smaller and smaller subsets while an associated decision tree is incrementally developed at the same time [28]

To build a decision tree, two types of entropy need to be calculated using frequency tables. Entropy refers to the probability distribution of the information contained in each observation (gain). Thus, the main RF algorithm steps in Biau [29] show that after calculating the entropy of the observations, the dataset is then split into the different attributes (trees). In choosing the attribute with the largest information gain as the decision node (root) and as the left node, which has an entropy of 0, the remaining nodes require further splitting. Thus, the algorithm is run recursively on the non-leaf branches until all data are classified [29].

Various methods exist for evaluating the quality of algorithm prediction to guarantee the selection of the best-performing classification algorithm. Among these are [30]:

- Confusion matrix (CM): shows the number of accurate and inaccurate predictions that the classification model makes compared to the actual outcomes (actual value) in the dataset.
- Receiver Operating Characteristic (ROC): also used for evaluation. ROC is a chart that shows a false positive rate (1-specificity) on the X-axis against a true positive rate (sensitivity) on the Y-axis.
- The area under the curve (AUC): determined by calculating the area under ROC curves; the quality of the classification model is measured, where the AUC should be between (0.5 and 1). When the area is close to one, it means that the classifier performance is acceptable; otherwise, if the area is less than 0.5, then the classifier performance is unacceptable because the classifier cannot distinguish between classes [31].

IV. The Proposed Model

A. Hardware and Software

Applying machine learning to classification becomes easier with the development of depth cameras and sensors to provide more accuracy in identifying the individual body parts of a naturally looking human [32]. Sign language relies on different body parts, which necessitates the use of multiple sensors. In this research, Kinect™ and Leap Motion Controller (LMC) sensors were used to create a model for recognizing ArSL gestures. Microsoft Kinect Version 2.0—which Microsoft released—has a Red Green Blue (RGB) depth camera and a human skeletal tracking algorithm that offers information about human body joints [33]. Meanwhile, LMC Version 2.0 provides a skeletal-tracking algorithm that offers information about hands and fingers as well as overall hand-tracking data, even if the hands cross over each other. "Figure 2." presents the 11 joints that needed to be retrieved via Kinect and the 12 points that needed to be retrieved via LMC in this research.

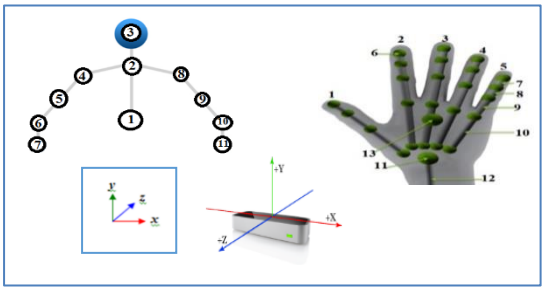


Fig. 2. Depth sensors' joint points detect based on Cartesian coordinate system

The Microsoft Kinect and LMC open-source software development kit (SDK) library were used to develop the proposed prototype with options for reading and managing visual depth information [34]. Visual Studio 2013 with C# was also used to calibrate the two devices, and the SQL Server Management Studio 2010 was used to create a relational database.

B. Data Collection

A prototype system was developed to collect data using the two sensors. The main window interface in the prototype provides real-time joint detection by representing the user's joint points as well as a histogram to give visual sign indications.

As participants gesture each letter they can individually click a button to save the body pose for each gesture.

"Figure 3" provides an example of a three-dimensional (3D) human skeleton where a line between each corresponding point was drawn (vector). To standardise the distance or depth metrics between the two devices, the length of each vector was converted from meters—which Kinect uses—to millimetres, which LMC uses, to standardise the length units in millimetres.

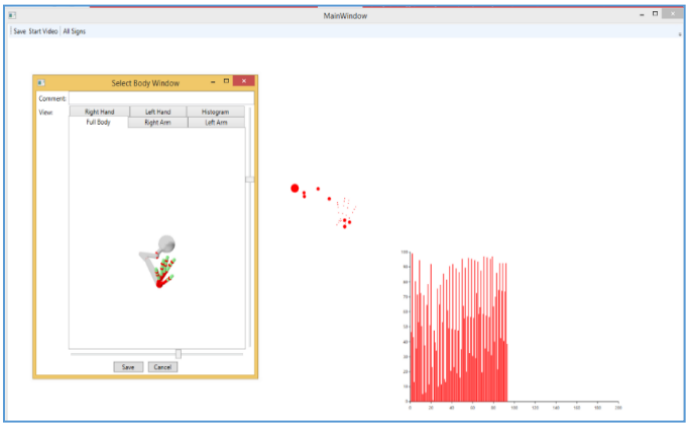


Fig. 3. Window in the prototype

Windows Media 3D from the Microsoft Development Network (MSDN) was used to visualise the captured data in the 3D space of human body joints by drawing one skeleton from the details retrieved from the two devices.

C. Feature Extraction

A feature represents a piece of information in any multimedia type, such as image, text, and video. It could be the direction of a certain object, such as the hand bones' direction [39]. For this research, the depth values that the two sensors captured were used to create two feature types, as seen in "Figure 4." Type one was denoted as "H" in the database; it included three angles for each hand bone, which were angles between the bone and the three axes of the coordinate system (X,Y,Z). Type two was denoted as "A" in the database; it included one angle between each of the two bones.

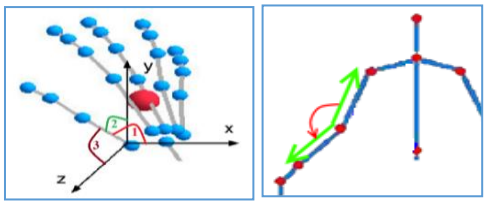


Fig. 4. Example of three angles for one joint (three angles) and one angle between two bones

These angles are the main factor for a comparison between the two gestures. Then, the prototype was considered ready to use in the experimental environment, as seen in "Figure 5." Twenty participants were asked to gesture the 28 Arabic alphabet letters. Each participant stood in front of the devices, which were connected to a personal computer, and he or she made around 28 to 40 gestures and mimicked sign gestures spanning seven days. Around 200 right gestures were collected daily for different letters from different participants.

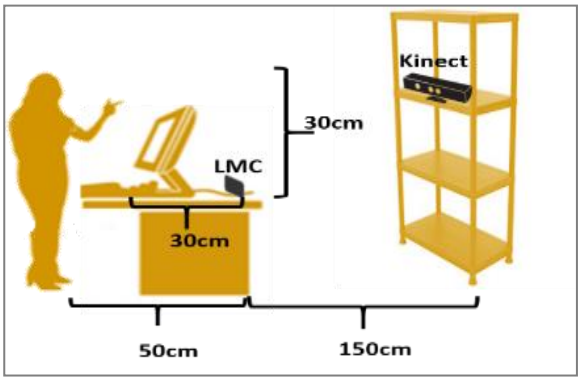


Fig. 5. The test environment with dimensions

Therefore, the number of gestured letters (observations) also varied between participants; for example, some participants gave five or more gestures for a specific letter. Table 1 shows the number of observations for each class (letter) in descending order.

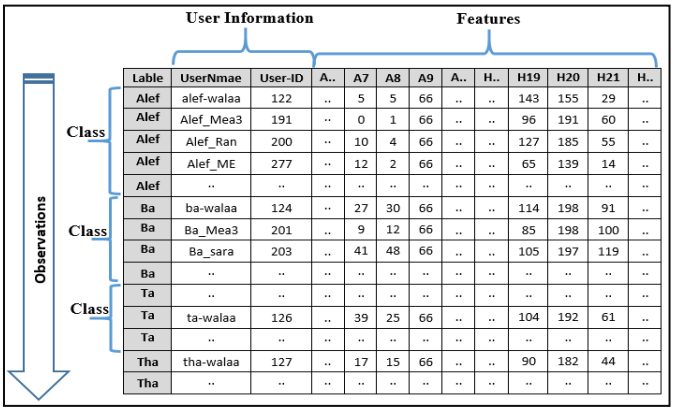
TABLE I. NUMBER OF OBSERVATIONS IN EACH CLASS

Class Name	Observations#	Class Name	Observations #	Class Name	Observations#
Ta	79	Shien	60	Waw	37
Kaf	74	Dal	55	Fa	36
Ba	71	Ha	48	Zay	36
Jiem	70	Alef	47	Ya	34
Sien	70	He	47	Ghayn	29
Qaf	68	Noon	47	Thal	19
Lam	68	Sad	47	Tah	14
Ra	68	Tha	45	Thah	12
Dhad	65	Ayn	44		
Miem	64	Kha	44		

The collected dataset had 235 features, presented in "Figure 6" as columns: the values of H0 to H180 were from type one, and the values of A1 to A54 were from type two. The dataset was reduced by selecting the body parts on which each gesture relied while removing all values that would not affect the interpretation of the ArSL letters. For example, the feature "A1" was an angle between the shoulder and right hand and would not affect the recognition of any ArSL letter depending on the hand bones only (at this point, the features became 102 values). In addition, the features with zero variance were removed; for example, when the variance of all values in feature "A9" was calculated, the result was zero, so that did not affect the recognition either (the features became 90 values).

The dataset observations are presented in "Figure 6" as rows, which include 1456 observations. Certain observations were removed as well, such as: 1) the rows that had the same values and 2) the rows that had multiple missing values (null values, where the device did not capture observation values well). The dataset was cleaned out for the 90 features' values, and the number of observations was changed to 1398.

"Figure 6" shows the dataset structure, where each observation was considered a letter from a specific participant and contained many features.



Lable	UserNmae	User-ID	A..	A7	A8	A9	A..	H..	H19	H20	H21	H..
Alef	alef-walaa	122	..	5	5	66	..	..	143	155	29	..
Alef	Alef_Mea3	191	..	0	1	66	..	..	96	191	60	..
Alef	Alef_Ran	200	..	10	4	66	..	..	127	185	55	..
Alef	Alef_ME	277	..	12	2	66	..	..	65	139	14	..
Alef	..	..	..	..	..	..	..	..	..	..	..	..
Ba	ba-walaa	124	..	27	30	66	..	..	114	198	91	..
Ba	Ba_Mea3	201	..	9	12	66	..	..	85	198	100	..
Ba	Ba_sara	203	..	41	48	66	..	..	105	197	119	..
Ba	..	..	..	..	..	..	..	..	..	..	..	..
Ta	..	..	..	..	..	..	..	..	..	..	..	..
Ta	ta-walaa	126	..	39	25	66	..	..	104	192	61	..
Ta	..	..	..	..	..	..	..	..	..	..	..	..
Tha	tha-walaa	127	..	17	15	66	..	..	90	182	44	..
Tha	..	..	..	..	..	..	..	..	..	..	..	..

Fig. 6. Original dataset structure

D. Classification Implementation

A dataset of 1456 gestured letters (observations) of the ArSL was collected. This original dataset was passed through three individual classifiers:

- SVM, which gave the highest accuracy results of ArSL letter classification in the experiments [13]
- RF, which many researchers recommend for its high accuracy [36]
- kNN, which is commonly used for its ease of interpretation and low processing time [25]

The results of the three classifiers were combined, and results were reused as a new dataset to train the same classifiers. The result of this combination is called an "ensemble schema dataset." Therefore, the training datasets were classified as an original dataset and an ensemble schema dataset.

The stacking schema was used for this research with only the classifiers' predictions (class labels were the letter names) as input for the ensemble model, without the original data, as seen in "Figure 7."

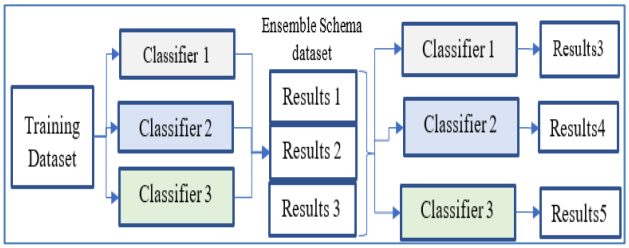


Fig. 7. Diagram of ensemble using stacking concept

In stacking's simplest form, the results from three different classifiers generated a new dataset named the "ensemble schema dataset." Classification passed two phases implementing the following steps:

1) The database was divided into two sets, training and testing set.

2) The training set was fed into classifiers to train them to recognise the class labels (letters).

3) The testing set was used to evaluate the classifiers' prediction ability (if it could recognise letters in the testing set accurately).

4) The CM showed the number of accurate and inaccurate predictions that the classifier made compared to the actual outcomes (actual value) in the testing set. Then, all classifiers' performance was evaluated by calculating the area under the ROC curves.

The implementation details of each phase are as follows in "Figure 8" and "Figure 9":

Phase 1: The raw database of 1456 observations (considered the letters) became 1398 after removing the rows that had the same values. The dataset was separated into a splitting ratio of a 75% to 25% training set with 1047 observations and a testing set with 351 observations. This training set was divided once more with the same splitting ratio into observations, such that:

- by using 730 observations, the model was trained to learn individually along with the right letter; and
- by using 317 observations, the model had to predict (classify) letters using the SVM, kNN, and RF algorithms.

Then, the prediction results from all three algorithms (317 predictions for each) were combined to become the training set of the three classifiers in phase 2. In addition, by using 351 observations, the model had to predict (classify) letters using the SVM, kNN, and RF algorithms as well. Then, the prediction results were combined to become the testing set of the three classifiers in phase 2.

Phase 2: The prediction data produced from phase 1 were used for the training step and then for the testing set of the 351 observations.

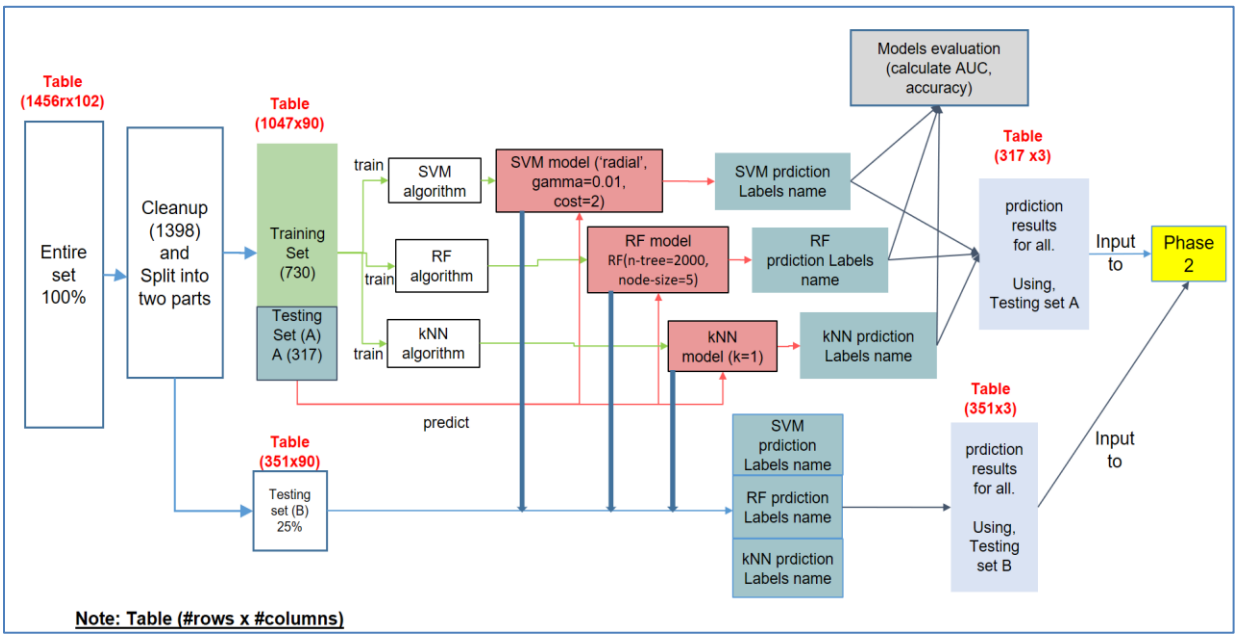


Fig. 8. Proposed model implementation structure in phase 1

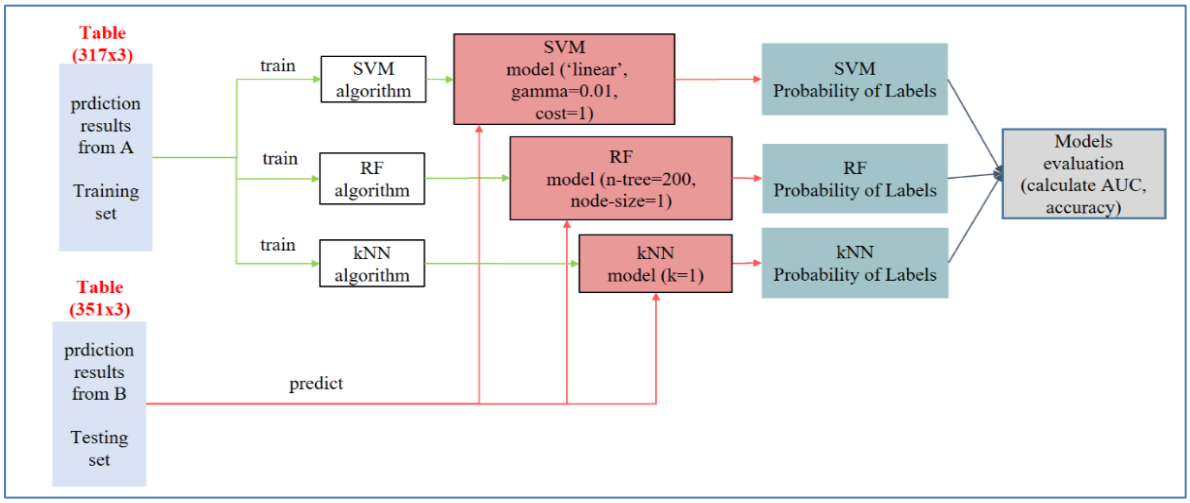


Fig. 9. Proposed model implementation structure in phase 2

E. Classification Results

The results of the classification in phase 1 for each classifier in detail are (Table 2):

1) kNN's parameter k was assigned a value equal to the square root of the available total number of observations. The value of k could be adjusted from 1 to 10. The value of k=1 was chosen for less computation and an accuracy of 85.484%.

2) SVM's two parameters—cost and gamma—were set to 2 and 0.01, respectively, to get the highest accuracy. In addition, kernel = "radial" because it uses curves instead of straight lines to separate the different labels; accuracy was 88.803%.

3) RF's two parameters: n(tree) (total number of trees to build) was set to 2000 and the node size (maximum children each tree can have) was set to five, which achieved an accuracy of 86.809%.

The results of the classification in phase 2 for each classifier in detail are (Table 2):

1) kNN had an accuracy of 87.151%, where the parameter k=1.

2) SVM had an accuracy of 86.880%, where the kernel = "linear"; and SVM's two parameters, cost and gamma, were set to 1 and 0.01, respectively.

3) RF had an accuracy of 88.048%, with RF's two parameters of n (tree) and node size, set to 200 and 1 respectively.

TABLE II. OVERALL ACCURACY

Classifier	Phase 1, Original dataset	Phase 2, Ensemble dataset
kNN	85.484%	87.151%
SVM	88.803%	86.880%
RF	86.809%	88.048%

The classifiers' performance in the two phases was evaluated using AUC for individual letter accuracy; these results are shown in "Figure 10" and "Figure 11."

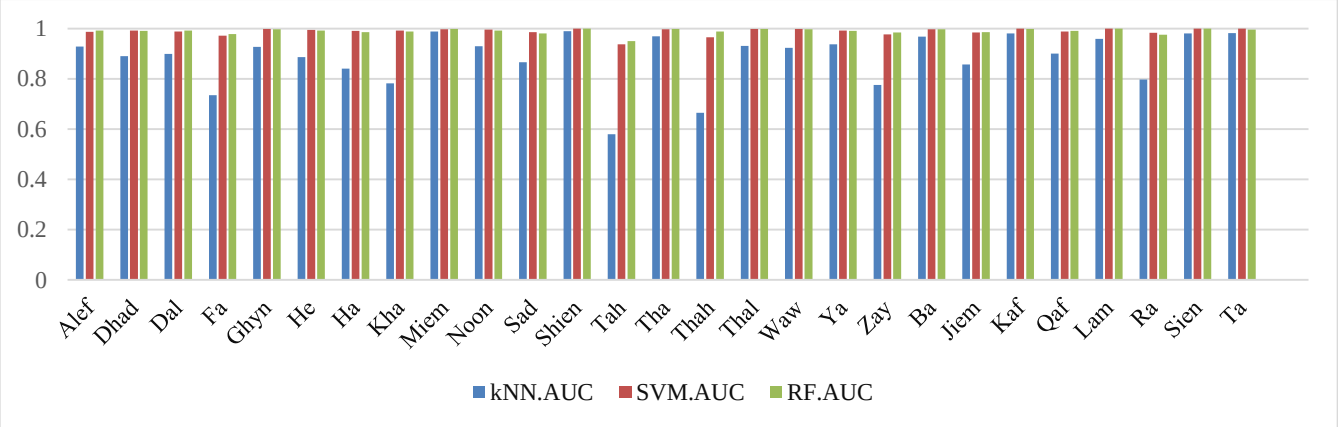


Fig. 10. The area under curve (AUC) for each classifier in phase 1

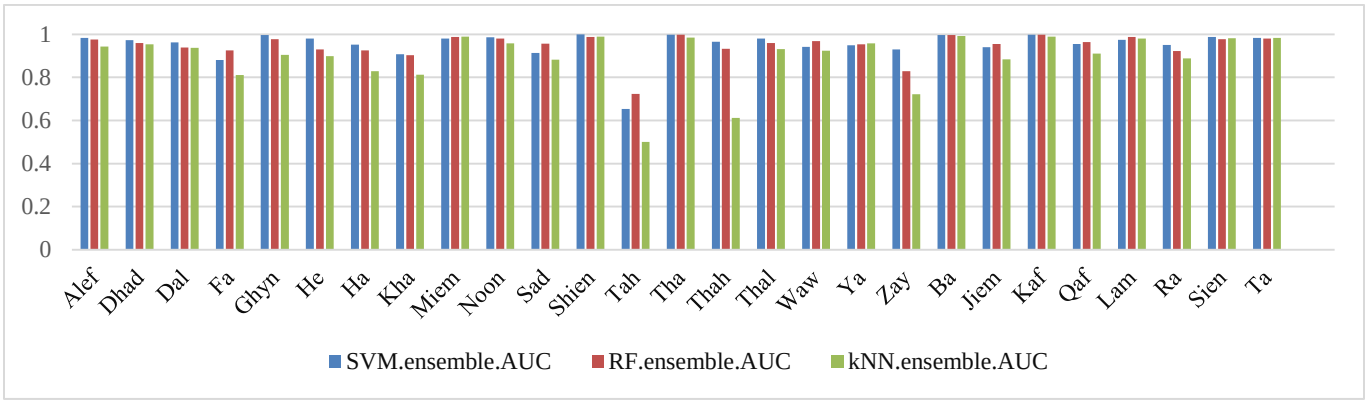


Fig. 11. The area under curve (AUC) for each classifier in phase 2

SVM achieved optimum results for this experiment when trained on the original dataset and not on the ensemble schema dataset, and this could be attributed to the variation in the number of observations for each class (Table 1). However, the devices had a low-speed response compared to human movement and a low precision of capturing the frames of a specific gestured letter. This was especially true for complex letters such as the following: ذ (Thal), ط (Tah), and ظ (Thah), where fingers overlapped, and the participant had to repeat the gesture or drop it altogether. This ultimately resulted in the variance between the numbers of observations for each class.

The variations in observation numbers were examined to assess if they affected the results. The discrepancy between the overall results of the algorithms used was investigated when it was trained on the original dataset and on the ensemble schema

dataset. The researchers proposed that the SVM could have achieved better results when applied to the original dataset due to the variance between the numbers of observations for each class. This was sometimes less than 10 in the training set, such as ذ (Thal), ط (Tah), and ظ (Thah), which had fewer than 20 observations. Running the three classifiers on these observations could have affected the overall results.

The three classifiers, kNN, SVM, and RF, were run on classes that had more than 65 observations each. The selection of 65 as a number is statically justified because the observations for each class were divided into training and testing sets, with the former requiring no fewer than 50 observations so that the model—which was based on the training set—would be satisfactory. In this particular case, it covered the highest observations under eight classes (letters), which are as follows: ت (Ta) with 79 observations, ك (Kaf) with 74 observations, ب (Ba) with 71 observations, ج (Jiem) with 70 observations, س (Sien) with 70 observations, ق (Qaf) with 68 observations, ل (Lam) with 68 observations, and ر (Ra) with 68 observations.

Moreover, the three classifiers (kNN, SVM, RF) were also re-run on the remaining 20 classes with fewer than 65 observations. Table 3 and Table 4 demonstrate the discrepancy noted earlier, which shows how the classifiers have changed in their overall accuracy results.

The eight ArSL letters that had more than 65 observations for each letter were analysed (Table 3). It was concluded that all of the classifiers' performance was enhanced when using a high number of observations. The accuracy results in phase 1 for kNN, SVM, and RF were 93.566%, 96.119%, and 93.846%, respectively. The results in phase 2 for kNN, SVM, and RF were 95.524%, 94.336%, and 95.699%, respectively.

TABLE III. CLASSIFICATION OVERALL ACCURACY 8 CLASSES (OBSERVATION NUMBERS >65)

Classifier	Phase 1, Original dataset	Phase 2, Ensemble dataset
kNN	93.566%	95.524%
SVM	96.119%	94.336%
RF	93.846%	95.699%

The remaining 20 classes of the ArSL Arabic alphabet, which had fewer than 65 observations for each letter, were also analysed (Table 4). The accuracy results in phase 1 for kNN, SVM, and RF were 85.216%, 88.221%, and 86.178%, respectively, and in phase 2, the results were 87.163%, 87.500%, and 88.413%, respectively.

TABLE IV. CLASSIFICATION OVERALL ACCURACY 20 CLASSES (OBSERVATION NUMBERS <65)

Classifier	Phase 1, Original dataset	Phase 2, Ensemble dataset
kNN	85.216%	87.163%
SVM	88.221%	87.500%
RF	86.178%	88.413%

Recognition accuracy results for each phase is as follows (Table 5):

1) Among individual classifiers, overall, SVM had higher accuracy in phase 1.

2) For the ensemble approach, overall, RF had higher accuracy in phase 2.

3) For all classes and classes with more than 65 observations, SVM had a higher accuracy in phase 1 than RF did in phase 2.

4) RF achieved higher accuracy in phase 2 for classes with fewer than 65 letters compared to SVM in phase 1, but the difference was negligible.

TABLE V. ALL RESULTS IN PHASE 1: ORIGINAL DATASET AND PHASE 2: ENSEMBLE DATASET

	All classes		8 classes > 65		20 classes < 65	
	Phase 1	*Phase 2*	*Phase 1*	*Phase 2*	*Phase 1*	*Phase 2*
kNN	85.484%	87.151%	93.566%	95.524%	85.216%	87.163%
SVM	88.803%	86.880%	96.119%	94.336%	88.221%	87.500%
RF	86.809%	88.048%	93.846%	95.699%	86.178%	88.413%

V. DISCUSSION AND CONCLUSION

This research used two depth sensors to capture all upper human skeleton joints, upon which most sign-language gestures rely. The supervised machine learning algorithms of kNN, SVM, and RF classified the depth values of gestures representing all ArSL letters.

It is essential to enhance the recognition accuracy of ArSL when using a supervised machine-learning approach, as it is important to get more accurate recognition results while avoiding complexity schema (the ensemble needs results from the three classifiers to classify the dataset), which requires more computation time.

The classification was performed using R packages, where three classifiers, SVM, kNN, and RF, were used to implement the general classification implementation process in two phases to recognise and interpret incoming gestures. In phase 1, the three classifiers of kNN, SVM, and RF were trained on the original dataset, whereas, in phase 2, the three classifiers were trained on an ensemble dataset, where the results of these three classifiers were combined into an ensemble schema dataset to classify the classes again. In addition, the various numbers of observations for each letter were analysed to check if various numbers affected the classifiers' accuracy performance.

As shown in Table 5, the recognition accuracy results were different among the three classifiers and among the two phases and for the different number of observations (classes with all observations, classes with fewer than 65 observations, and classes with more than 65 observations).

The researchers concluded that the implementation of SVM produced a higher overall accuracy when running as an individual classifier, no matter the number of observations. However, for small datasets, RF's ensemble approach could be used, as it had higher accuracy than SVM did in phase 1.

Although RF produced higher accuracy results for classes with limited class observation data, the difference between the accuracy results of RF in phase 2 and SVM in phase 1 was negligible. Such a difference does not warrant using an ensemble approach, which adds more processing complexity, as required with the ensemble approach.

With such a result, SVM used as an individual classifier would be the more efficient choice because it produces higher recognition accuracy with less complexity.

Future work on this subject could address how this prototype can be used to collect and classify dynamic gestures (multiple frames) that represent the sign of one word or phrase.

REFERENCES

[1] A. A. Youssif, A. E. Aboutabl, and H. H. Ali, "Arabic Sign Language (ARSL) recognition system using HMM," Int. J. Adv. Comput. Sci. Appl. IJACSA Location: The Science and Information (SAI) Organization, vol. 2, no. 11, pp. 45 - 51, 2011.

[2] I. E. Naqa and M. J. Murphy, "What is machine learning?," in Machine Learning in Radiation Oncology, I. E. Naqa, R. Li, and M. J. Murphy, Eds. Location: Springer International Publishing, 2015, pp. 3–11.

[3] A. Munoz, "Machine learning and optimization," Courant Inst. Math. Sci, 2014. [Online]. Available: https://www.cims.nyu.edu/~munoz/files/ml_optimization.pdf. [Accessed: 26-Mar-2017].

[4] P. K. Pisharady and M. Saerbeck, "Recent methods and databases in vision-based hand gesture recognition: A review," Comput. Vis. Image Underst. Location: Computer Vision and Image Understanding, vol. 141, pp. 152–165, 2015.

[5] [5] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," IEEE Trans. Pattern Anal. Mach. Intell. Location: IEEE, vol. 35, no. 8, pp. 1798–1828, August 2013.

[6] R. Begg and J. Kamruzzaman, "A machine learning approach for automated recognition of movement patterns using basic, kinetic and kinematic gait data," J. Biomech. Location: Elsevier Ltd, vol. 38, no. 3, pp. 401–408, March 2005.

[7] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," Comput. Vis. Image Underst. Location: Elsevier Inc, vol. 108, no. 1–2, pp. 52–73, October 2007.

[8] D. Ruta and B. Gabrys, "Classifier selection for majority voting," Inf. Fusion Location: Elsevier B.V, vol. 6, no. 1, pp. 63–81, 2005.

[9] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," Periodical Title Location: IOS press, vol.159, pp.3-24, 2007.

[10] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," Mach. Learn. Location: Kluwer Academic Publishers, vol. 51, no. 2, pp. 181–207, 2003.

[11] R. Polikar, "Ensemble based systems in decision making," IEEE Circuits Syst. Mag. Location: IEEE, vol. 6, no. 3, pp. 21–45, 2006.

[12] [12] L. Rokach, "Ensemble-based classifiers," Artif. Intell. Rev. Location: Springer link and ProQuest Technology Collection, vol. 33, no. 1–2, pp. 1–39, February 2010.

[13] M. A. Almasre and H. Al-Nuaim, "Recognizing Arabic Sign Language gestures using depth sensors and a KSVM classifier," in 2016 8th Computer Science and Electronic Engineering (CEEC), 2016, pp. 146–151.

[14] R. E. Schapire, "A brief introduction to boosting," in Ijcai, vol. 99, Editors' Information, Eds. Location: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999, pp. 1401–1406.

[15] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," J.-Jpn. Soc. Artif. Intell. Location: Journal-Japanese Society For Artificial Intelligence, vol. 14, no. 771–780, p. 1612, 1999.

[16] C. D. Sutton, "Classification and regression trees, bagging, and boosting," in Handbook of Statistics, vol. 24, E. J. W. and J. L. S. C.R. Rao, Eds. Location: Elsevier, 2005, pp. 303–329.

[17] L. Li, Q. Hu, X. Wu, and D. Yu, "Exploration of classification confidence in ensemble learning," Pattern Recognit. Location: Elsevier B.V, vol. 47, no. 9, pp. 3120–3131, September 2014.

[18] M. Farooq and E. Sazonov, "Detection of chewing from piezoelectric film sensor signals using ensemble classifiers," in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2016, pp. 4929–4932.

[19] M. Woźniak, M. Graña, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," Inf. Fusion Location: Elsevier B.V, vol. 16, pp. 3–17, March 2014.

[20] D. Wolpert, "Stacked generalization (stacking)," Machine Learning, 2017. [Online]. Available: http://machine-learning.martinsewell.com/ensembles/stacking/. [Accessed: 26-Mar-2017].

[21] C. J. Burges, "A tutorial on support vector machines for pattern recognition," Data Min. Knowl. Discov. Location: ProQuest Technology Collection, vol. 2, no. 2, pp. 121–167, 1998.

[22] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," J. Mach. Learn. Res. Location: The Journal of Machine Learning Research, vol. 2, pp. 125–137, 2002.

[23] "Learning kernels SVM," R-bloggers, 25-Sep-2012.

[24] S. McCann and D. G. Lowe, "Local Naive Bayes Nearest Neighbor for image classification," in 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3650–3656.

[25] S. Cost and S. Salzberg, "A weighted nearest neighbor algorithm for learning with symbolic features," Mach. Learn. Location: Springer, vol. 10, no. 1, pp. 57–78, 1993.

[26] A. Dhurandhar and A. Dobra, "Probabilistic characterization of nearest neighbor classifier," Int. J. Mach. Learn. Cybern. Location: Springer Berlin Heidelberg, vol. 4, no. 4, pp. 259–272, August 2013.

[27] C.-L. Chang, "Finding prototypes for nearest neighbor classifiers," IEEE Trans. Comput. Location: IEEE, vol. C-23, no. 11, pp. 1179–1184, November 1974.

[28] L. Breiman, "Random forests," Mach. Learn. Location, vol. 45, no. 1, pp. 5–32, 2001.

[29] Găš. Biau, "Analysis of a random forests model," J. Mach. Learn. Res. Location: Journal of Machine Learning Research, vol. 13, pp. 1063–1095, April 2012.

[30] S. Sayad, "Model evaluation," An Introduction to Data Mining, 2017-2010. [Online]. Available: http://www.saedsayad.com/model_evaluation_c.htm. [Accessed: 14-Aug-2016].

[31] C. Ferri, J. Hernández-Orallo, and M. A. Salido, "Volume under the ROC surface for multi-class problems," in European Conference on Machine Learning, 2003, pp. 108–120.

[32] D. Ionescu, V. Suse, C. Gadea, B. Solomon, B. Ionescu, and S. Islam, "An infrared-based depth camera for gesture-based control of virtual environments," in Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2013 IEEE International Conference, 2013, pp. 13–18.

[33] A. Jana, Kinect for Windows SDK Programming Guide: Build Motion-Sensing Applications with Microsoft's Kinect for Windows SDK Quickly and Easily. Birmingham: Packt Publ., 2012, pp. 40–50.

[34] M. A. Almasre and H. Al-Nuaim, "A real-time letter recognition model for Arabic Sign Language using Kinect and Leap Motion Controller v2," IJAEMS Open Access Int. J. Infogain Publ. Location vol. 2, no. 5, pp. 514–523, 2016.

[35] D. M. Gavrila, "The visual analysis of human movement: A survey," Comput. Vis. Image Underst. Location: Elsevier Inc, vol. 73, no. 1, pp. 82–98, January 1999.

[36] R. Su, X. Chen, S. Cao, and X. Zhang, "Random forest-based recognition of isolated sign language subwords using data from accelerometers and surface electromyographic sensors," Sensors Location: MDPI AG, vol. 16, no. 1, p. 100, January 2016.