Open access • Journal Article • DOI:10.1007/S12530-016-9169-1

An Artificial Immune System for Offline Isolated Handwritten Arabic Character Recognition — Source link

Chaouki Boufenar, Mohamed Batouche, Marc Schoenauer

Institutions: University of Jijel, University of Paris-Sud

Published on: 01 Mar 2018 - Evolving Systems (Springer Berlin Heidelberg)

Topics: Artificial immune system, Feature extraction, Cursive and Supervised learning

An Artificial Immune System for Offline Isolated Handwritten Arabic Character Recognition

Chaouki Boufenar, Mohamed Batouche, Marc Schoenauer

Noname manuscript No.
(will be inserted by the editor)

An Artificial Immune System for Offline Isolated Handwritten Arabic Character Recognition

Chaouki Boufenar · Mohamed Batouche · Marc Schoenauer

Received: date / Accepted: date

Abstract Character recognition plays an important role in the modern world. In recent years, character recognition systems for different languages has gain importance. The recognition of Arabic writing is still an important challenge due to its cursive nature and great topological variability. The Artificial Immune System is a supervised learning technique that embodies the concepts of natural immunity to cope with complex classification problems. The objective of this research is to investigate the applicability of an Artificial Immune System in Offline Isolated Handwritten Arabic Characters. The developed system is composed of three main modules: preprocessing, feature extraction and recognition. The system was trained and tested with ten-fold cross-validation technique on an original realistic database that we built from the well-known IFN/ENIT benchmark. Parameter tuning was performed with a grid-search algorithm with leave-one-out cross-validation. The obtained results of the proposed system are promising with a classification rate of 93.25% and often outperform most well-known classifiers from Scikit Learn Library.

Keywords Isolated handwritten Arabic character · Offline recognition · Artificial Immune System · Preprocessing · Feature extraction · IFN/ENIT · Scikit Learn Library

1 Introduction

Handwriting recognition has been one of the most fascinating and challenging research areas in the field of image processing and pattern recognition during the recent years [39]. Offline handwritten character recognition systems are very important for the creation of electronic libraries, mail sorting, checks verification, to mention a few examples. Arabic characters are used by several languages such as Persian, Shahmukhi, Urdu and Jawi. However, compared to Latin and Chinese handwritten character recognition, little research has been done on handwritten Arabic character (HAC) recognition.

Nature is an immense source of inspiration for solving hard and complex problems in computer science. Recently, bio-inspired algorithms have explored new areas of application and more opportunities in computing [62, 74].

C. Boufenar
Computer Science Department, University Of Jijel, 18000 Jijel , Algeria
Tel.: +213 (0) 34-50-18-65
Fax: +213 (0) 34-50-14-00
E-mail: boufenarc@gmail.com

M. Batouche
Computer Science Department, College of NTIC, University Of Constantine2, 25000 Constantine , Algeria
Tel.: +213 (0) 31-78-31-65
Fax: +213 (0) 31-78-31-65
E-mail: mohamed.batouche@univ-constantine2.dz

M. Schoenauer
TAO Team, INRIA Saclay–Île-de-France, LRI, University of Paris-Sud and CNRS,
91405 Orsay Cedex, France
Tel.: +33 (0)1 69-15-66-26
Fax: +33 (0)1 69-15-42-13
E-mail: marc.schoenauer@inria.fr

The vertebrate immune system (IS) is one of the most intricate bodily systems and its complexity is sometimes compared to that of the brain [10]. A simplistic view of the immune system (IS) is that of an organ whose function is to detect and protect the organism from dangerous invaders called antigens.

Artificial Immune System (AIS) is inspired from the Natural Immune System (NIS), which uses antibodies (recognition cell or B-Cells) to recognize antigens. The term used to describe the degree of similarity between B-Cells and an antigen is called affinity. The adaptive ability of the IS is a process called affinity maturation. During an immune response, the recognition cell will perform clonal expansion; it will generate many clones of itself in an attempt to gain a better match next time the antigen will be found. A process called somatic hypermutation mutates the generated clones in proportion to the affinity [63].

Based on those principles, interesting artificial immune algorithms were developed. Among them, we note the resource limited AIS proposed by Timmis and Neal [61]. This algorithm introduced the concept of an artificial recognition ball (ARB), which has the same meaning as a B-cell.

In 2001, Watkins introduced the artificial immune recognition system (AIRS) which is a bio-inspired classification algorithm [66]. An improvement denoted AIRS2 was made in 2004 by Watkins et al. [65]. The AIRS is a supervised learning technique that embodies the concepts of natural immunity to cope with complex classification problems. The AIRS is a resource limited algorithm that imitates immune metaphors such as antibody-antigen binding, affinity maturation, clonal selection process, resource competition and memory cell acquisition.

Many researchers try to apply several techniques for breaking through the complex problems of handwritten character recognition [52]. Compared to bio-inspired approaches such as artificial neural networks (ANN) [38], genetic algorithms [30, 38, 43], AIS approaches remained less explored. There are some works on recognizing handwritten characters in English [36], Nepali [71], Armenian [69], Kannada [37], Russian and Oriental scripts [68]. However, there are only a few reports on Arabic script.

To the best of our knowledge, AIRS has not been used for offline isolated HAC recognition. It is believed that this is due to the lack of a standard benchmark database for comparing and validating emerging approaches in HAC research field. Motivated by these ideas, we are interested in applying an AIRS-based system to an original database for offline handwritten Arabic character recognition. For this purpose, we have set three different goals:

1. Providing an original dataset for isolated offline handwritten Arabic character recognition.
2. Investigating the applicability of AIRS to our new dataset.
3. Improving the performance of AIRS on this application by using a cross-validation grid-search parameter tuning.

The remaining of the paper is organized as follows. In the first section, we give a brief overview on some significant recent contributions to HAC recognition. Section 3 gives a state of the art regarding the use of AIS-based algorithm in handwritten recognition field. Section 4 presents the proposed system and Section 5 describes our new dataset used in this study. Section 6 describes the experiments conducted and compares their results with other leading work. Finally, Section 7 presents the conclusions and some future research directions.

2 Handwritten Arabic Character Recognition: Related works

In the last few decades, an increasing body of research have been conducted on Arabic character recognition. For more detailed overviews about Arabic character recognition, the readers may refer to reviews [34, 59].

Jannoud [26] proposed an Automatic Arabic Handwritten Text Recognition System. After the segmentation stage, discrete wavelet transform was used to extract the features of each character. A minimum distance classifier was used to classify the characters. This system achieved a 90% recognition rate of characters.

To recognize isolated HAC, Abandah et al. [1] extracted a subset of 40 features from 95 features using principal component analysis. Classification was done by applying the following classifiers: Quadratic Discriminant Analysis (QDA), Linear Discriminat Analysis (LDA), Diagonal QDA (DQDA), Diagonal LDA (DLDA) and KNN (K-Nearest Neighbors). The experiments were conducted on their own database which contains 4992 characters. On average, accuracy of 87% was achieved in the best case.

Aljuaid et al. [4] used a genetic approach to recognize HAC. Structural features of each character are extracted to distinguish the shape of the character. The system was tested and achieved an accuracy of approximately 87%.

Al-Jawfi [3] designed the structure of LeNet neural network, which recognizes the main and secondary components. The network configuration had more than three layers which increased the computation and complexity of the network. The mean square error averaged was 0.42 on test set. The database used to train and test the network consists of 750 segmented Arabic characters which is not enough to show the strengths of the ANN model.

El-Glaly and Quek [17] extracted some statistical features depending on the number of vertical and horizontal transitions and the ratio between the height and width of the characters. KNN and ANN classifiers were compared in this work. Contrary to ANN, KNN had very low error rate in classifying new datasets, and its accuracy was 90%. However, the comparison between the two classifiers using few simple examples (280 samples) fails to show the strength of the classifiers.

Rashad et al. [54] used a combination of statistical features and geometric moment features, which are independent of the font and size of the character. An ANN was trained 1000 times using 1000 samples of characters. The results showed that using only statistical features was less accurate than using a hybrid features systems. However, the execution time is high because their ANN architecture included more than two layers. An average recognition rate of 97% is achieved using six different fonts on 1000 samples.

Sahlol and Suen [56] introduced a new method for HAC recognition based on novel preprocessing operations and also different features. Their system was trained and tested by ANN on CENPRMI dataset. The proposed algorithm obtained good results as it was able to recognize 88% of their test set accurately.

Elzobi et al. [18] extracted the mean and standard deviation of the Gabor wavelet transformed images. This method uses the Gabor wavelet which requires memory and high computation time. Classification was carried out by employing a Support Vectors Machine (SVM) algorithm where IESK-arDB and IFN/ENIT databases were respectively used for testing and evaluating the proposed approach. The average recognition rate reached 71%.

Bahashwan et al. [7] proposed an efficient algorithm for HAC recognition. The algorithm combined features extracted from curvelet transform [9] and spatial domains. The extracted feature vector was then trained using an ANN classifier. The evaluation was carried out on their own database which contains 5600 samples. An accuracy rate of 90.3% has been achieved.

In reviewing works on the recognition of HAC, the following limits were raised:

1. The SVM classifier has not been able to recognize the characters in IESK-arDB and IFN/ENIT databases, since the recognition rate was very low. It was not advantageous for HAC recognition.
2. The best results were found using the ANN, but these results are not significant, since the ANN was trained and tested on little data.

Despite these efforts, most of the recognition rates obtained in private datasets are suboptimal and need to be improved.

3 Artificial Immune Systems for Handwritten Recognition

AIS approach is applied to several applications [13], it aims at solving complex computational or engineering problems such as pattern recognition [10, 15, 16, 19, 24, 33, 72] computer virus detection and elimination [27, 67], and optimization [28].

AIRS has been proposed to medical diagnosis problems in [32, 47–51]. It was also used for semantic documents classification in [22], music genre classification in [21] and in [29] for genre and author detection in Turkish texts.

Concerning handwritten recognition, Garain et al. [20] proposed a classification approach based on CSA to recognize handwritten Indian digits. Two different datasets were used to evaluate the proposed classification approach. The training was conducted on samples from five partitions and classification was tested on the sixth partition. The experimental results reported the average recognition accuracy of about 96%.

Chmielewski and Wierzchoń [12] proposed an immune-based approach to recognition of handwritten words. They employed the negative selection mechanism and incorporated two types of detectors : binary and real values. A dataset of few samples was used in their experiments. To evaluate the effectiveness of the algorithm, detection rate and false alarm rate were used. The results showed the efficiency of the proposed approach.

Chenet al. [11] proposed a handwritten character recognition algorithm based on AIS. The authors used the well-known character dataset provided by F. Prat from University of California at Irvine. The dataset were divided into two sets: 6500 samples for training and 1500 for testing. The average recognition accuracy of the algorithm was 94.41%.

Yu Yang used AIS to recognize Russian uppercase character [68], handwritten Icelandic character [70], handwritten Armenian character [69] and Nepali character [71]. They used in their study 1920 samples of 32 Icelandic characters and achieved an average accuracy of 83.2%, 81.2% and 80.7% respectively. The contrast experiments were done using ANN and indicated that AIS has more advantages than ANN (75.1%) when applied to a small datasets.

In order to enhance the accuracy, a letter recognition algorithm based on AIS, referred to as LEBAI, was presented by Liang et al. [36]. The algorithm was tested by the well-known letter recognition dataset of UCI (University of California at Irvine) and achieved a recognition accuracy of 95.58%.

Huang et al. [23] presented a new Licence Plate Character Recognition (LPCR) algorithm based on clonal selection algorithm. Once memory cells are established, it will output the classification results using Fuzzy KNN approach. The performance of the algorithm was compared to the ANN in solving a LPCR problem and showed to be better by more than two percent in training and more than three percent in testing.

Utpal et al. [20] presented an application of a 2-phase clonal selection algorithm for the recognition of handwritten Indic numerals. The proposed scheme achieved a recognition accuracy of about 96%.

Mamatha et al. [37] used an AIS for training zonal extracted features of handwritten Kannada numerals. KNN classifier was used for classification. The performance of the proposed algorithm was investigated in detail on nearly 1250 samples of handwritten Kannada numerals and a recognition accuracy of 98.11% was achieved.

Nemmour and Chibani [42] examined the application of AIRS for solving handwritten Arabic word recognition system. The AIRS was tested on samples of the IFN/ENIT [44] dataset using ridgelet transform [8] and grid features. The performance evaluation was carried out comparatively to SVM classifiers.

In recent time, Arit et al. [60] proposed a new character recognition system for archaic Lanna handwritten characters. The CSA of the AIS was hybridized with particle swarm optimization and used to build a recognition model. The experiments were conducted on printed Lanna character, Bangla numerals, Arabic numerals, Devanagari numerals, Telugu numerals and Latin ones.

Besides, Serdouk et al. [57,58] introduced an improved AIRS classification for solving automatic offline handwritten signature verification. The results showed that AIRS for handwritten signature verification has promising performance and outperforms the state of the art algorithms.

Based on the literature review of the AIRS classifier and compared with other languages, the following points are drawn:

1. The AIRS technique has not been used for offline isolated HAC recognition.
2. The tuning of AIRS parameters has not been mentioned.

We present in the following the methodology that we have adopted for developing our system. The proposed system is based on an artificial immune approach for recognizing isolated HAC.

4 The proposed character recognition system

The system that we propose is composed of three main modules: prepocessing module, feature extraction module and recognition module. We have developed the first and second modules with Matlab whereas we used Java and Weka to develop the third one. The programs can be accessed through this link: https://www.mediafire.com/folder/74kzyuevvb8jz/Documents. Fig. 1 shows the flowchart of the system.

4.1 Preprocessing module

We perform the following preprocessing steps:

4.1.1 Noise removal

The major objective of noise removal is to remove any unwanted pixel-patterns which do not have any significance in the output. We perform three operations:

1. Filling: fills isolated interior pixels, such as the center pixel in the pattern below (see Fig. 2(a)).

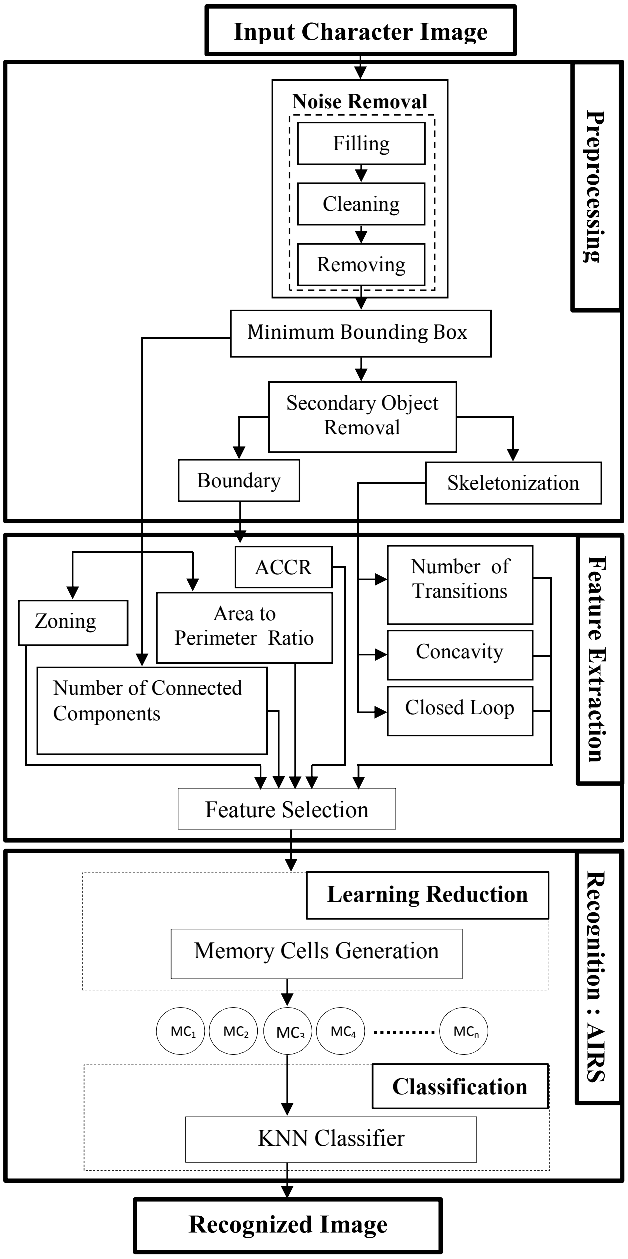


Fig. 1 Proposed recognition scheme for isolated HAC.

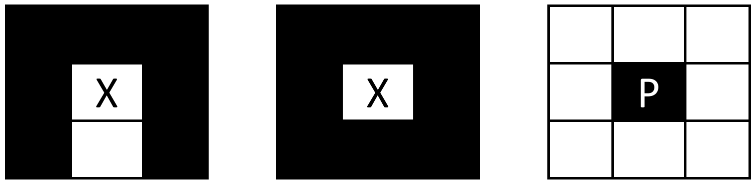
2. Cleaning: removes isolated pixels, such as the center pixel in the pattern below (see Fig. 2(b)).
3. Removing: removes isolated pixels (individual black pixels that are surrounded by white ones), such as the center pixel in the pattern below (see Fig. 2(c)).

4.1.2 Minimum Enclosing Bounding Box

It consists of computing the smaller rectangular frame (bounding box) enclosing the character. This operation is shown in Fig. 3(d).

4.1.3 Skeletonization

It is a morphological operation, somewhat related to erosion or opening. This module reduces all lines to have single pixel thickness. Such an operation allows recognizing the character image using relatively few patterns in the training set. Fig. 3(d) shows the resulting outline of Zhang and Suen thinning algorithm [73] on isolated Dal(د) character.



(a) Filling. (b) Cleaning. (c) Removing.

Fig. 2 Morphological noise removal.

(a) Input character image. (b) Skeletal representation. (c) Contour representation. (d) Minimum Enclosing Bounding Box.

Fig. 3 Preprocessing operations.

4.1.4 Boundary extraction

It is another preprocessing stage in character recognition where the character outer contour is found. Fig. 3(c) shows isolated Dal(د) after finding its boundary.

4.1.5 Secondary Component Removal

We use some features which are extracted from the main bodies skeleton and boundary of the character. Thus, removing secondary component is done before starting the feature extraction process. Indeed, more than half the Arabic letters are composed of main body and secondary components. The secondary components are letter components that are disconnected from the main body. These letters can be detected by using the connected component labeling techniques [25, 55].

4.2 Feature extraction module

4.2.1 Feature extraction

The process of getting useful information from the character image is called *feature extraction*. Choosing the most discriminant features might be the most important step for achieving a high recognition rate. In general there are two categories of features extracted: structural features and statistical features. We have implemented seven types of features distributed as follow:

1) Structural information. number of connected components, closed loop and concavities.
2) Statistical information. zoning, accumulated chain code representation, area to perimeter ratio and transition features.

- Zoning
  This technique imposes a grid of 4×4 on the bounding box of character image dividing the image into 16 equal zones. In each zone, we extract the density that represents the ratio of the number of black pixels forming the character on the total size of a zone [41]. In such a way , we will have a vector of 16 real values. In addition, we compute the horizontal, vertical, both diagonal and anti-diagonal histograms. However, we consider zones instead of pixels. Thus, the feature vector has 26 features (16 for densities and 10 for histograms). Fig. 4 illustrates zoning features.

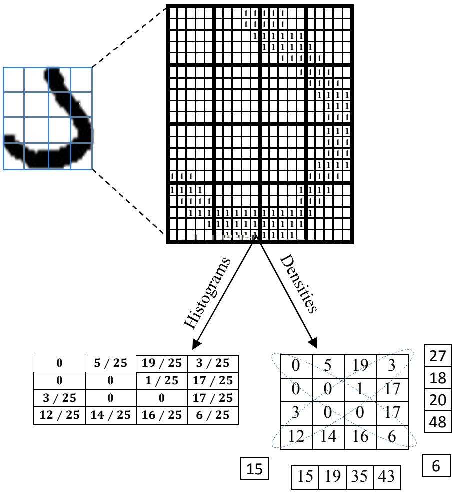
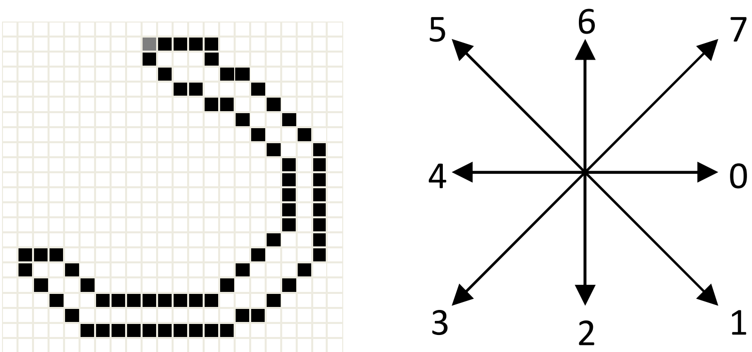


Fig. 4 Zoning Features.



(a) Image boundary.

(b) 8-Freeman Chain Code.

0000210111112222222233334344444444455556001110000000777776666555545456

(c) Freeman Chain Code Sequence.

(14, 9, 8, 5, 12, 11, 6, 5)

(d) Accumulated representation.

Fig. 5 Accumulated Freeman Chain Code Representation.

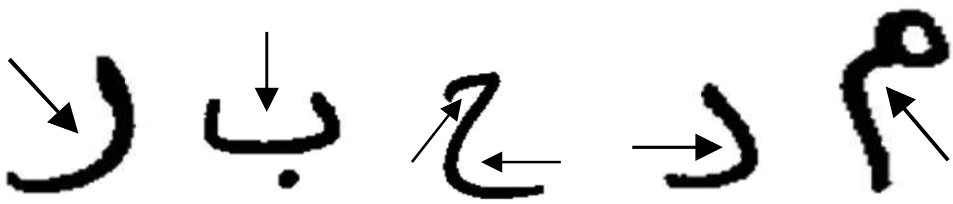
- Accumulated Chain Code Representation (ACCR)
  Chain code representation of an input character image shown in Fig. 5(c) is obtained by using Freeman codes while tracing its contour as shown in Fig. 5(a). Accumulate representation shown in Fig. 5(d) consists of browsing the contour of the image pixel by pixel and accumulate the number of pixels in the same direction given by the 8-Freeman code as shown in Fig. 5(b). As we use 8-connectivity, we will have a feature vector composed of 8 components representing the accumulated numbers.
- Transition features
  For each row and each column in the binary image, we compute the number of transitions between foreground pixels and background pixels ((1 to 0) or (0 to 1)). Then, the largest values in the two directions constitute the transition features vector. Thus, this features contribute with two components in the global features vector. A computation example is shown in Fig. 6.
- Concavity
  Arabic characters have strokes concaved in different directions, e.g, up in Baa character (ب), right in Haa(ح) character, left or up-left in Raa(ر) character and down-left in Meem(م) character. This

3	1	1	0	1	0	0
3	0	1	1	1	0	1
4	0	1	0	1	1	0
5	**0**	**1**	**0**	**1**	**0**	**1**
3	1	1	0	1	1	0
	2	0	2	0	3	**4**

Fig. 6 Transition Features.

(a) Original shape with three secondary components.
(b) Variant with two Secondary Components.
(c) Variant with one Secondary Component.

Fig. 7 Variations in Arabic handwritten secondary components.



(a) Up-Left.
(b) Up.
(c) Right and down-left.
(d) Left.
(e) Down-Right.

Fig. 8 Types of Concavities.

feature contributes with four components in the global feature vector. Fig. 8 shows five types of concavity.

- Closed loop
  There are nine Arabic letters that are usually written with loop (ص, ض, ط, ظ, ف, ق, م, ه and و). This makes the closed loop feature a prominent part in recognizing Arabic characters. We used the connected component labeling technique [25, 55]to detect loops in an image. Thus, this feature contributes with one component in the global features vector.
- Area to perimeter ratio
  Since different people write the same characters in different sizes, the absolute area and perimeter are not suitable features for HAC. Therefore, the ratio (area/perimeter ratio) of the character is more appropriate. This feature participates with one real component in the global features vector.
- Number of connected components
  The number of connected components is an important feature because it can be effective for distinguishing among most of Arabic characters like Baa(ب) with one dot, Taa(ت) with two dots, Thaa(ث) with three dots and Alef_Hamzah(أ) containing Hamzah(ء) as secondary component. There are im-

Table 1 Feature Vector.

N.°	Feature	Size
f1	Accumulated chain code representation	8
f2	Concavity	4
f3	Zoning	26
f4	Number of connected components	1
f5	Transition features	2
f6	Area to Perimeter Ratio	1
f7	Closed loop	1

Table 2 Mapping between the Immune System and AIRS.

Immune System	AIRS
Antibody	Attribute (feature) vector
Recognition ball	Combination of feature vector and vector class
Antigens	Training instances
Clonal expansion	Reproduction of ARBs (Artificial Recognition Balls) that are well matched with antigens
Affinity maturation	Random mutation of ARBs and removal of the least stimulated ARBs
Immune memory	Memory set of mutated ARBs

portant variations in drawing the secondary components that can be confusing; mostly in drawing two dots and three dots. Calculating the size of component is the most suitable solution for this kind of confusion. Thus, this feature participates with one component in the global features vector. Fig. 7 shows variations in shapes of the same Arabic handwritten letter Thaa(ث).

The global feature vector includes seven sets of features which are summarized in Table 1.

4.2.2 Feature selection

To select the most relevant features and reduce the dimension of a feature vector, we apply a deterministic feature selection technique. In fact, in this work we have selected the categories of features but not the elements in the category. Let $S = \{f_i\}$ be the set of these categories of features, $i \in \{2, 3, 4, 5, 6, 7\}$ (see Table 1). We consider a combination C_n^m in S, where $m \in \{2, 3, 4, 5, 6, 7\}$ and $n = 7$.

4.3 Recognition: AIRS

The function of the AIRS algorithm is to prepare a pool of recognition or memory cells (data exemplars) which are representative of the training data that the model is exposed to, and is suitable for classifying unseen data. In this study, antibodies are character samples which have been trained and antigens are unseen characters which will be recognized (input samples).

Before exposing the learning and classification algorithms relative to the AIRS technique, it is interesting to show the relationship between NIS and AIRS. Table 2 summarizes the mapping from IS to AIRS [65].

The AIRS algorithm is basically composed of two phases: the learning reduction phase and the classification phase. Fig. 9 shows the life cycle of the AIRS system.

4.3.1 The learning reduction phase

The learning reduction phase represents the main step of the algorithm. It consists of four main stages. In AIRS, there are two different populations: Artificial Recognition Balls (ARBs) and Memory Cells (MC).

In AIRS, the ARB corresponds to the feature vector of a training sample in addition to its class and its number of resources.

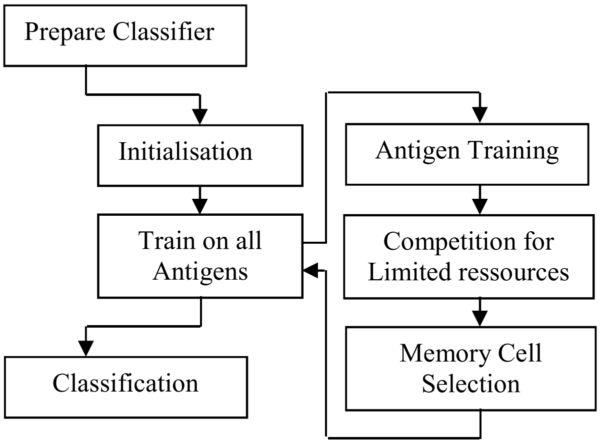


Fig. 9 Life cycle of the AIRS system.

We assume the AB as the ARBs population: $AB = \{ab_1, ab_2, ..., ab_r\}$ and $MC = \{mc_1, mc_2, ..., mc_m\}$ as memory cell pool containing m memory cells.

- Initialization
  This step of the algorithm consists of preparing the data for use in the training process. The training data is normalized so that the affinity of each pair (ag_i, ag_j) in the training set is in the range$[0, 1]$. The Euclidean distance is used as affinity measure between two cells or antigens:

$$affinity(ag_i, ag_j) = \sqrt{\sum_{k=1}^{na} (ag_i(k) - ag_j(k))^2}.$$

  where (ag_i, ag_j) is a pair of antigens (instances) and na represents the number of attributes. After normalization, the affinity threshold is calculated. The Affinity Threshold (AT) is the average affinity value over all training data. The affinity threshold is calculated as follows:

$$AffinityThreshold(AT) = \frac{\sum_{i=1}^{n} \sum_{j=i+1}^{n} affinity(ag_i, ag_j)}{\frac{n \times (n-1)}{2}}.$$

  Where n is the number of training instances (antigens) in question. Finally, the MC is seeded by randomly adding training instances. Once initialization is complete, the following three steps will repeat for each antigen in the training base.
- Step 1: Memory cell identification and ARB generation
  This step models the principle of affinity maturation and takes place for each new antigen introduced. Given a specific training antigen ag, find the memory cell, mc_{match}, that has the highest stimulation with ag.

$$mc_{match} = argmax_{mc \in MC} Stimulation(ag, mc).$$

  where $stimulation(ag, mc)$ is defined as follow:

$$Stimulation(ag, mc) = 1 - affinity(ag, mc).$$

  This sub-step is called Clonal Expansion. Once mc_{match} is selected, it will be used to generate new ARBs to be placed into the population of (possibly) preexisting ARBs pool AB. The number of mutated clones to be added is proportional to the affinity between mc_{match} and ag. This number is calculated by:

$$numClones = Stimulation(ag, mc) \times HypermutationRate \times ClonalRate.$$

  Note that HR and CR are values set by the user. Features of new clones can be mutated with probability mutationRate which is also a userdefined constant in range[0,1].
- Step 2: Competition for limited resources and development of a candidate memory cell
  The goal of this stage is to determinate a candidate memory cell ($mc_{candidate}$). This is determined as follows: For each $ab \in AB$

1. Calculate Stimulation Value ($StimV$) depending on whether it belongs or not to antigen class.
2. Calculate the number of resources (NbR) to allocate based on $StimV$:

$$NbR(ab) = StimV(ab) \times ClonalRate.$$

3. The resources are split across all ARBs as follows: 50% on those belonging to the same class of antigen and the remaining half on the rest of ARBs.
4. After attribution of resources, those ARBs left with zero rewards are removed from the pool of ARBs (AB).
5. At this level, a new ARBs pool will appear. All surviving ARBs of a class are cloned, mutate and placed back in AB. The number of clones is obtained by the following formula:

$$NbClones = StimV \times ClonalRate.$$

6. Stopping condition : The process stops when the mean $StimV$ over all remaining ARBs (those with $StimV$ different to zero value) of a class is above a threshold value, namely, the Stimulation Threshold (ST) which is set by the user.

- Step 3: $mc_{candidate}$ selection
  Select the ARB with highest stimulation in the antigen class. This ARB is the candidate memory cell($mc_{candidate}$). If the $mc_{candidate}$ is more stimulated by ag than mc_{match}, it will be added to the pool. Moreover, if the affinity between $mc_{candidate}$ and mc_{match} is less than Affinity Threshold (AT: calculated in the initialization stage) times the ATS (user defined parameter), then mc_{match} is removed and replaced by $mc_{candidate}$. Finally, the ARB pool is reset to zero after each antigenic pattern.

4.3.2 The classification phase

Once the training of all antigens is completed, the evolved MC pool can be used for classifying test samples. Specifically, classification is accomplished by majority vote of the k-nearest memory cell to the presented test antigen (k is user parameter).

5 DataBase

Large databases of HAC and words are confidential and not publicly available for non-commercial research when compared to Latin languages. Many papers in HAC recognition have used their own small datasets such as Lawgali et al. [35], Al-Badr and Mahmoud [2], Khedher et al. [31], Mozaffari et al. [40], Bahashwan et al. [6] and recently Jawad H Alkhateeb [5]. Most cited databases are confidential and not freely available. The few available ones, are small and the characters are not evenly distributed over all classes. In this study, we have constructed our local database that contains 5600 isolated letters from the well-known IFN/ENIT benchmark [45].

The process operates as follows:

– First, we have considered the first three sets in IFN/ENIT database (set_a, set_b and set_c).
– Secondly, we have picked up a word in a current set, and we have cropped it manually in letters using the paint software.
– Finally, we have kept only isolated letters which we will be recognized.

We summarize by saying that we have provided a new local dataset that includes 5600 isolated black and white letters of 128 × 128 pixels. The images are evenly divided into 28 classes, the variability is quite important as we can assess from the dataset mosaic shown on Fig. 10. At first glance, the dataset seems similar to MNIST dataset [14] in terms of learning complexity. The provided dataset can be downloaded through this link: https://www.mediafire.com/folder/74kzyuevvb8jz/Documents

6 Experiments and discussions

The evaluation of the AIRS model was performed in three steps: feature selection step, parameter tuning step and comparison step. At first, we run AIRS with default parameter values : ($Seed = 1, ATS = 0.2, CR = 10, HR = 2.0, TR = 150, ST = 0.9, MIPS = 1, k\_neighbor = 3$).

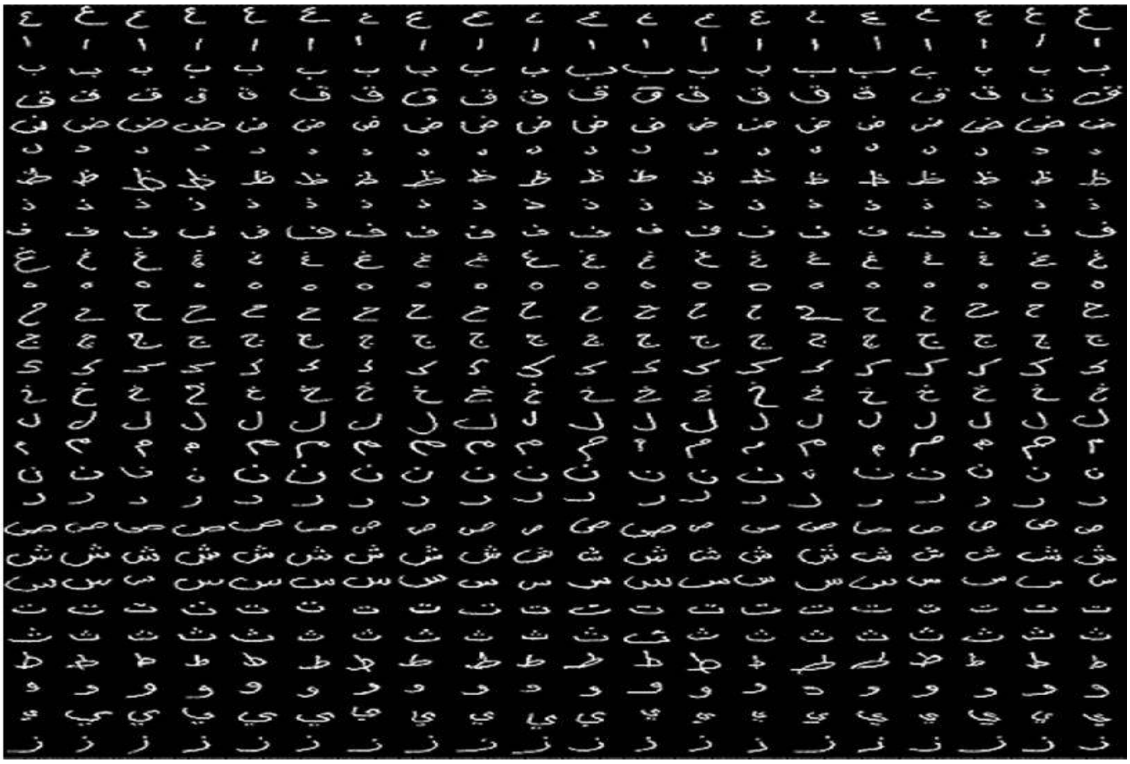


Fig. 10 Partial view of the dataset (each line represents a class).

6.1 Feature selection step

We have considered for the experiment the seven sets of features described in section 4.2. Selecting the most relevant features has been done by running AIRS with default values over 120 combinations of seven sets of features and with leave-one-out ten-cross-validation. We have obtained the highest accuracy with combination $(f_3, f_4, f_5, f_6, f_7)$ that corresponds to zoning, number of connected components, transition features, area to perimeter ratio and closed loop.

6.2 Parameter tuning step

Parameters tuning is a final step in the process of applied machine learning before presenting results. Algorithm tuning has been recently addressed in some research works on different applications [53, 64].

In fact, the classification performance of the AIRS2 algorithm depends on eight user defined parameters: Seed, Affinity Threshold Scalar (ATS), Clonal Rate (CR), Hypermutation Rate (HR), Total Resources (TR), ST, Memory Initial Pool Size (MIPS), number of nearest neighbors in KNN(K). Investigating individually the effects of parameters allows one to reduce the number of parameters which were varied within grid-search algorithm. For tuning parameters we considered three levels:

A) Global Investigation Level.
At this level we have examined the impact of all parameters in large ranges. Fig. 11 shows how the accuracy is affected by alternating parameters.

- Varying a number of seed cells (Seed) and MIPS between one and 100 affects the classification performance. Here it can be noted that increasing the Seed and MIPS has a very little influence on the performance of AIRS algorithm. Thus, these two parameters were kept constant with the following values: Seed=1 and MIPS=1.
- Varying ATS between 0.05 and one with step=0.05 showed that the accuracy decreases from the first value. This needs to examine deeply the effect of this parameter between 10^{-2} and 10^{-1} with step=10^{-2}.
- How CR parameter affects classification performance is shown in Fig. 11. When we have varied CR between one and 100 with step =5, a little impact on accuracy has been observed between

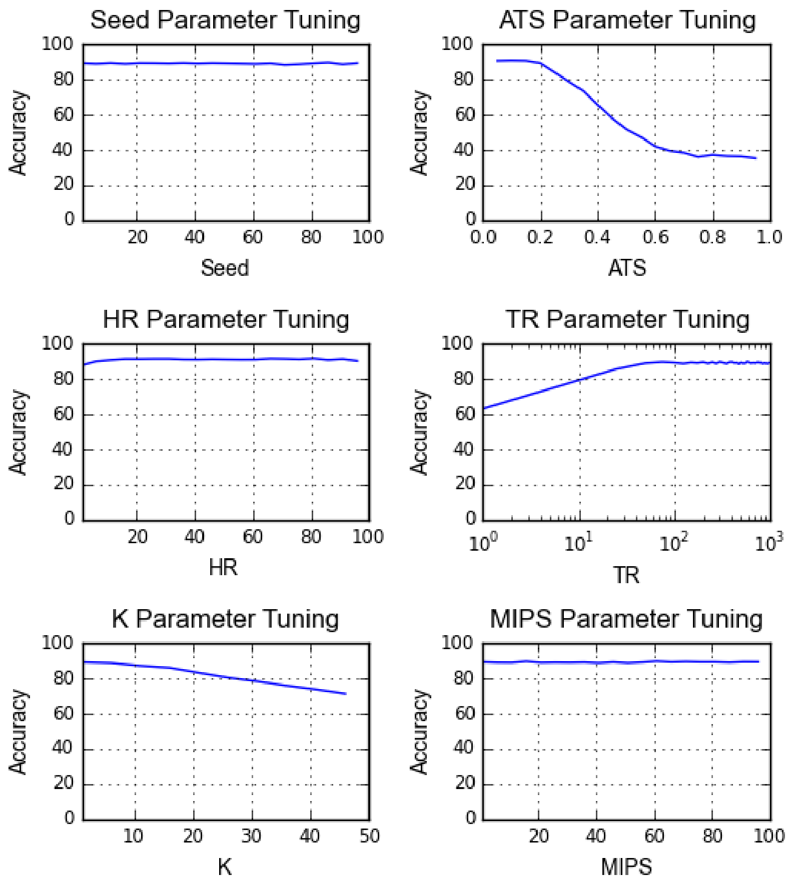


Fig. 11 The effect of altering parameter values on classification accuracy: global investigation.

one and 10. Over CR=10, the accuracy keeps its highest value which is close to 91% until CR=50. When the CR value exceeds 50, the performance of AIRS begins to degrade until CR=85 when we have shown a serious decrease in accuracy which reaches its bad values. Consequently, CR value has been more explored between 10 and 40 using a smaller step (step =0.5).

– Changing HR between one and 100 with step=5 has shown an improvement beyond HR=10. Better value has been reached from HR=5, and accuracy remains unaffected by variation in HR parameter until 100 value. This parameter has been more investigated in the vicinity of five using steps of 10^{-1}.
– TR parameter when varied between one and 10^3 with step=25 accuracy drops significantly from 63.05% for TR=1 to 89.67% for TR=75. However, beyond TR=75 classification accuracy has not exceeded 89.9% which has been the highest value. More exploration has been needed in the range $[75, 200]$ with step=5 to better tune this parameter.
– Varying the ST parameter between 10^{-1} and one with step=0.15 has shown that the accuracy benefits slightly from higher threshold. Over ST=0.6 the accuracy becomes almost constant between 89% and 89.5%. Therefore, a deeper exploration has been conducted in the range $[0.6, 0.98]$ using 10^{-2} as step.
– Varying the number of the nearest neighbor (k) between one and 50 has shown a continued deterioration in the performance of the algorithm. The first value of k gives the best accuracy. Thus, k has been fixed to value *one*, and will not be varied in the next step.

B) Local Investigation Level
The purpose of this step is to further reduce the number of parameters to be varied in the next step. We will make an exploration more precise than that shown in the previous step. Table 3 and Fig. 12 summarize the variations of parameter values in local ranges.

- **TR** and **ST** : For both two parameters the accuracy was stagnating at 89% for all values looked into. They have a very little impact on the performance of the model when they deviate from

Table 3 Parameter values variation in local ranges.

Parameter	Value
Seed	1
MIPS	1
ATS	Explored in $[10^{-2}, 10^{-1}]$, step=10^{-2}
CR	Explored in $[10, 40]$, step=0.5
HR	Explored in $[2, 15]$, step=0.5
TR	Explored in $[75, 200]$, step=5
ST	Explored in $[0.6, 0.98]$, step=10^{-2}
K	1

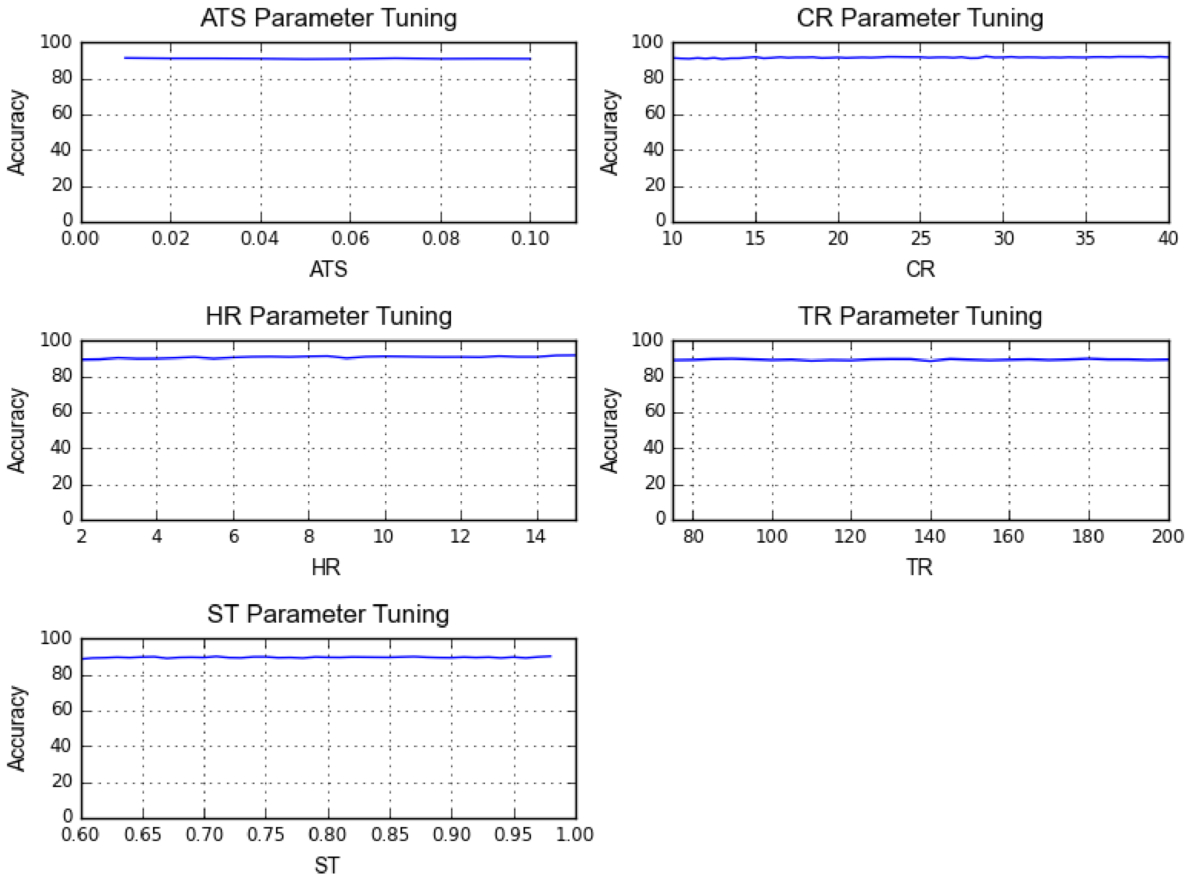


Fig. 12 The effect of altering parameter values on classification accuracy: local investigation.

their default values (0.9 and 150, respectively). Therefore, TR and ST will maintain their default values and they will not be among the parameters to which we apply the grid-search algorithm.

- **A**TS and **H**R: in the overall investigated intervals the accuracy is close to 91%. Thus, these two parameters can improve accuracy when they are correlated.
- **C**R: investigation of this parameter in the range $[10, 40]$ has allowed us to observe that:
  - The accuracy does not drop below 91% on all values of CR examined in the interval $[10, 40]$;
  - The best predictions ($\sim$ 92%) are obtained for the values close to CR=40;

  The three last parameters (ATS, HR and CR) will undergo a grid-search to show the effect of correlating them.

C) Grid-Search Tuning Level.
At this level we have applied the grid-search algorithm with ten leave-one-out cross-validation in order to find the optimal combination of parameters values of ATS, CR and HR, such that the best

Table 4 Grid-search parameter optimization details.

Classifier	Parameter	Range $[Min, Max]$	Steps	Scale	Optimized parameter	Accuracy
AIRS	ATS	$[10^{-2}, 10^{-1}]$	3	Logarithmic	10^{-2}	93.25%
	CR	$[30, 40]$	6	Logarithmic	35.652	
	HR	$[6, 15]$	6	Logarithmic	12.488	

classification accuracy on our dataset can be achieved. The search range for these parameters was $[10^{-2}$,$10^{-1}]$, $[30, 40]$ and $[6, 15]$, respectively. For both parameters, the grid-search has been performed with steps on a logarithmic scale. Table 4 shows the grid-search parameter optimization details while Table 5 illustrates the recognition results by character class.

We show in Table 5 that AIRS gives a recognition rate exceeding 90% for almost all letters although we get 100% success rate for some letters.

The worst six recognized letters are: (ج), (ح), (خ), (ع), (غ), (ق). The first five are always drawn with concavities. There are subtle similarities among the three first and the only difference between them is the presence of secondary components (points in this case) and their positions with respect to the main component. It has been observed from Table 5 that most of the ambiguities in recognizing occur between the characters with very similar shapes. Examples of some incorrectly recognized characters are the following:

– Character Khaa(خ) is confused with character Ghayn(غ), especially when the little right cavity of character Ghayn(غ) is omitted.
– The only character that can be confused with Zay(ز) character is the Dhal(ذ) character; but Raa(ر) character can be confused with three characters that are: Lam(ل), Dal(د), and Waw(و) notably when it is written without loop. For this reason we find that the accuracy of character Zay is 100%, while the accuracy of character Raa(ر) is 93.5% and the difference between these two characters is only the dot over Zay.
– Character Taa(ت) is very confused with character Thaa(ث) when the three dots above the latter are written as two dots. This is observed in the third entry in Table 5.

According to the obtained results in Table 5, we think that the model accuracy can be improved by introducing new features that:

– Detect the position of secondary components.
– Find the number of cavities and their directions.

6.3 Comparative Analysis

This section briefly touches on some comparisons in two ways, first between our system and recent state of the art recognition systems, and second with the well-known classifiers from Scikit Learn Library. In order to carry out these comparisons, we have developed a Python class that can be accessed through this link: https://www.mediafire.com/folder/74kzyuevvb8jz/Documents

Table 6 shows the recognition results of recent published systems and our system. For each work, the table shows its publication year, the used classifier, the database used for evaluation and its accuracy. We compare our work to that of Abandah et al. [1] and Elzobi et al. [18] which segment the word into characters and then recognize it. The former applies multiple classifiers to a local database of 4992 samples, and the latter uses SVM classifier on IESK-arDB database. In addition, our system is compared to those of Al-Jawfi [3] and El-Glaly and Quek [17] which obtain a better accuracy with KNN classifier but on very few examples (758 and 280, respectively). We also compare our work with Sahlol and Suen [56] and Bahashwan et al. [7] which use ANN classifier to recognize handwritten Arabic characters. Unlike the former, the latter uses a local database of the same size as ours.

Table 6 shows that the proposed system is well situated among the best recently published works with an accuracy of 93.25%, which can be considered as a highly promising result.

Table 5 Recognition rate for each character in the proposed system.

Character Class	Number of correctly recognized images	Details of misrecognized images (Character : Number of images)						Accuracy (%)
Alef(ا)	198	(ع:1)	(ف:1)					99
Baa(ب)	199	(ف:1)						99.50
Taa(ت)	186	(ث:2)	(خ:1)	(ن:10)	(ص:1)			93
Thaa(ث)	182	(ت:14)	(ز:1)	(ج:1)	(ن:1)	(ش:1)		91
Jeem(ج)	179	(ع:1)	(خ:8)	(ح:7)	(غ:5)			89.5
Haa(ح)	169	(ا:27)	(ك:4)					89.5
Khaa(خ)	150	(غ:42)	(ج:8)					75
Dal(د)	182	(ك:2)	(ا:2)					91
Dhal(ذ)	195	(ز:4)	(ن:1)					97.5
Raa(ر)	187	(د:2)	(و :2)	(ل:9)				93.5
Zay(ز)	200	0						100
Seen(س)	198	(ل:1)	(و:1)					99
Sheen(ش)	193	(ت:2)	(ث:4)	(ي:1)				96.5
Sad(ص)	190	(ض:4)	(م:1)	(ط:1)				95
Dad(ض)	186	(ق:1)	(ف:8)	(ص:5)				93
TTaa(ط)	192	(ص:3)	(و:3)	(ظ:2)				96
Dhaa(ظ)	182	(ض:8)						91
Ayn(ع)	168	(ح:26)	(خ:3)	(ك:3)				84
Ghayn(غ)	173	(ج:1)	(خ:26)					86.5
Faa(ف)	190	(ق:1)	(ض:4)	(ظ:1)	(و:1)	(ن:2)	(ص:1)	95
Qaf(ق)	171	(ض:5)	(ظ:2)	(ف:13)	(ن:1)	(ج:1)	(ت:7)	85.5
Kaf(ك)	197	(ح:1)	(ل:2)					98.5
Lam(ل)	182	(د:6)	(ذ:1)	(ر:11)				91
Meem(م)	197	(و:3)						98.5
Noon(ن)	194	(ت:3)	(ز:2)	(خ:1)				97
Ha(ه)	198	(م:1)	(و:1)					99
Waw(و)	196	(ر:2)	(ه:2)					98
Yaa(ي)	188	(ب:7)	(ج:1)	(خ:1)	(ك:2)	(ز:1)		94

In order to compare AIRS with other methods on the same database, we have considered eight classifiers from the well-known Scikit Learn Library [46]. We have run these classifiers using ten-fold cross-validation and *GridSearchCV*() as grid-search meta classifier for parameters optimization. A detailed list of the optimized parameters is provided in Table 7.

Table 8 shows that except for Random Forest classifier, optimized AIRS outperforms all optimized classifiers and is able to recognize with an interesting accuracy rate of 93.25%. For more transparency, we have compared AIRS with SVM classifier. The comparison results have illustrated that AIRS model has better accuracy than SVM classifier with linear kernel (92.31%) and rbf kernel (92.81%). We learn from this table that the AIRS model parameters affect its performance. We observe although that our parameters optimization strategy has allowed to improve the accuracy from 89.39% to 93.25%. Comparison with KNN shows well the importance of the learning-reduction step of AIRS model where the recognition rate was at 76.91%.

Table 6 Comparison with state of the art.

Authors	Year	Database(Samples)	Classifier	Accuracy
Adandah et al. [1]	2008	Local database(4992)	QDA, DQDA, LQDA, LDA, KNN	84%
Al-Jawfi [3]	2009	Local database(758)	KNN	99.08%
El-Glaly and Quek [17]	2012	Local database(280)	ANN KNN	60% 90%
Rashad et al. [54]	2012	not mentioned	ANN	97%
Sahlol and Suen [56]	2014	CENPRMI(11620)	ANN	88%
Elzobi et al. [18]	2014	IESK-arDB	SVM	71%
Bahashwan et al. [7]	2015	Local database(5600)	ANN	90.3%
Our work	2016	Local database (5600)	AIRS	93.25%

Table 7 The classifiers and their optimized parameters are listed. The grid-search attributes (Range, Steps and Scale) are shown for each parameter.

Classifier	Parameter	Range $[Min, Max]$	Steps	Scale	Optimized parameter
Random Forest	n-estimators	$[1, 20]$	1	Linear	18
Linear SVM	C	$[10^{-3}, 10^{3}]$	40	Logarithmic	1193776×10^{-6}
Rbf SVM	C γ	$[10^{-3}, 10^{10}]$ $[10^{-9}, 10^{5}]$	15 15	Logarithmic Logarithmic	22758459×10^{-2} 10^{-8}
K-Nearest Neighbor	K	$[1, 20]$	1	Linear	9

Table 8 Comparison with Scikit Learn Library classifiers on the same dataset and over the optimized parameters.

Classifier	Recognition Rate (%)
Random Forest (n_estimators)	95.60
AIRS (Optimized parameter)	**93.25**
Rbf_SVM(C,γ)	92.81
LDA()	92.61
Linear_SVM(C)	92.31
Decision Tree()	91.66
Naïve Bayes ()	90.67
AIRS(default parameter values)	89.39
QDA()	89.12
KNN (K) (without reduction learning phase)	76.91

7 CONCLUSION AND FUTURE WORK

In this paper, the use of AIRS in the field of offline HAC recognition has been investigated and discussed. We have proposed a novel system composed of three main phases: preprocessing, feature extraction and classification.

At first, we have performed some preprocessing operations like noise removal, skeletonization, boundary extraction and secondary component removal. At the second phase, some structural and statistical features have been extracted and feature selection step has been performed to retain the most relevant features. At the last phase, we have used the AIRS model as a classifier.

We have provided an original database for isolated offline handwritten Arabic character recognition to evaluate our system. This database includes 5600 characters built from the well-known IFN/ENIT benchmark. The images in our new database are cropped and resized in 128×128 pixels.

We have experimented our approach on the novel database and under the default parameters of the AIRS model. The first results of experiments were very promising and an accuracy of 89.39% was achieved.

In order to improve accuracy, we have addressed the potential problem of the large number of optimized parameters for the AIRS model. The optimization of parameters with cross-validation grid-search increased significantly the classification accuracy from 89.39% to 93.25%.

To allow an equivalent comparison, a set of classifiers from the well-known Scikit learn library has also undergone a parameters optimization step using a grid-search technique over range, steps and scale attributes. The AIRS model outperforms almost all classifiers using hand-crafted features, and showed a great efficiency.

This study offers a good contribution to the literature. It presents an efficient model in comparison with some recently published works in the state of the art. We see the following promising avenues for future work:

1. Look into the strength of AIRS on a large dataset of HAC.
2. Tune the parameters automatically using a parameter tuning algorithm.
3. Investigate the combination of AIRS model with other classification techniques.
4. Compare the AIRS to deep learning techniques such as Convolutional Neural Networks and Deep Belief Neural Networks when the features are automatically computed by the network.

References

1. Abandah, G.A., Younis, K.S., Khedher, M.Z.: Handwritten arabic character recognition using multiple classifiers based on letter form. In: Proceedings of the 5th IASTED International Conference on Signal Processing, Pattern Recognition, and Applications (SPPRA08), pp. 128–133. ACTA Press, Innsbruck (2008)
2. Al-Badr, B., Mahmoud, S.A.: Survey and bibliography of arabic optical text recognition. Signal processing **41**(1), 49–77 (1995)
3. Al-Jawfi, R.: Handwriting arabic character recognition lenet using neural network. Int. Arab J. Inf. Technol. **6**(3), 304–309 (2009)
4. Aljuaid, H., Mohamad, D., Sarfraz, M.: Arabic handwriting recognition using projection profile and genetic approach. In: 2009 Fifth International Conference on Signal Image Technology and Internet Based Systems, pp. 118–125. IEEE, Morocco (2009)
5. AlKhateeb, J.H.: A database for arabic handwritten character recognition. Procedia Computer Science **65**, 556–561. Elsevier, Prague Czech Republic (2015)
6. Bahashwan, M.A., Abu Bakar, S.A.: A database of arabic handwritten characters. In: Control System, Computing and Engineering (ICCSCE), 2014 IEEE International Conference on, pp. 632–635. IEEE, Malaysia (2014)
7. Bahashwan, M.A., Abu Bakar, S.A.: Offline handwritten arabic character recognition using features extracted from curvelet and spatial domains. Research Journal of Applied Sciences, Engineering and Technology **11**(2), 158–164 (2015)
8. Candès, E.J., Donoho, D.L.: Ridgelets: A key to higher-dimensional intermittency? Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **357**(1760), 2495–2509 (1999)
9. Candes, E.J., Donoho, D.L.: Curvelets: A surprisingly effective nonadaptive representation for objects with edges. Tech. rep., DTIC Document (2000)
10. de Castro, L.N., Timmis, J.: Artificial immune systems: a novel paradigm to pattern recognition. Artificial Neural networks in pattern Recognition **1**, 67–84 (2002)
11. Chen, Y., Liang, C., Yang, D., Peng, L., Zhong, X.: A handwritten character recognition algorithm based on artificial immune. In: Computer Application and System Modeling (ICCASM), 2010 International Conference on, vol. 12, pp. V12–273. IEEE, Taiyuan (2010)
12. Chmielewski, A., Wierzchoń, S.T.: An immune approach to recognition of handwritten words. In: Biometrics and Kansei Engineering, 2009. ICBAKE 2009. International Conference on, pp. 49–54. IEEE, Cieszyn (2009)
13. De Castro, L.N., Von Zuben, F.J.: Artificial immune systems: Part ii–a survey of applications. FEEC/Univ. Campinas, Campinas, Brazil (2000)
14. Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29**(6), 141–142 (2012)
15. Djeddi, C., Souici-Meslati, L.: Artificial immune recognition system for arabic writer identification. In: Innovation in Information & Communication Technology (ISIICT), 2011 Fourth International Symposium on, pp. 159–165. IEEE, Amman (2011)
16. Draman, N.A., Wilson, C., Ling, S.: Bio-inspired audio content-based retrieval framework (b-acrf). World Academy of Science, Engineering and Technology **29**(5), 785–790 (2009)
17. El-Glaly, Y., Quek, F.: Isolated handwritten arabic character recognition using multilayer perceptron and k nearest neighbor classifiers. Computer Science Department, Virginia Polytechnic Institute and State University, USA (2011)
18. Elzobi, M., Al-Hamadi, A., Al Aghbari, Z., Dings, L., Saeed, A.: Gabor wavelet recognition approach for off-line handwritten arabic using explicit segmentation. In: Image Processing and Communications Challenges 5, pp. 245–254. Springer, Switzerland (2014)

19. Faraoun, K., Boukelif, A.: Artificial immune systems for text-dependent speaker recognition. INFOCOMP Journal of Computer Science **5**(4), 19–26 (2006)
20. Garain, U., Chakraborty, M.P., Dasgupta, D.: Recognition of handwritten indic script using clonal selection algorithm. In: Artificial Immune Systems, pp. 256–266. Springer, Berlin Heidelberg (2006)
21. Golzari, S., Doraisamy, S., Sulaiman, M.N., Udzir, N.I.: A hybrid approach to traditional malay music genre classification: Combining feature selection and artificial immune recognition system. In: Information Technology, 2008. ITSim 2008. International Symposium on, vol. 2, pp. 1–6. IEEE, Malaysia (2008)
22. Greensmith, J., Cayzer, S.: An artificial immune system approach to semantic document classification. In: Artificial Immune Systems, pp. 136–146. Springer, Berlin Heidelberg (2003)
23. Huang, R., Tawfik, H., Nagar, A.: Licence plate character recognition using artificial immune technique. In: Computational Science–ICCS 2008, pp. 823–832. Springer, Cambridge (2008)
24. Isa, N., Sabri, N.M., Jazahanim, K.S., Taylor, N.K.: Application of the clonal selection algorithm in artificial immune systems for shape recognition. In: Information Retrieval & Knowledge Management,(CAMP), 2010 International Conference on, pp. 223–228. IEEE, Selangor (2010)
25. Jain, K., Rangachar, B.: Schunck. machine vision, pp. 1–549. McGraw-Hill New York (1995)
26. Jannoud, I.A.: Automatic arabic hand written text recognition system. American Journal of Applied Sciences **4**(11), 857–864 (2007)
27. Jim, L.E., Gregory, M.A.: A review of artificial immune system based security frameworks for manet. International Journal of Communications, Network and System Sciences **9**(1), 1 (2016)
28. Jordehi, A.R.: A chaotic artificial immune system optimisation algorithm for solving global continuous optimisation problems. Neural Computing and Applications **26**(4), 827–833 (2015)
29. Kaban, Z., Diri, B.: Genre and author detection in turkish texts using artificial immune recognition systems. In: Signal Processing, Communication and Applications Conference, 2008. SIU 2008. IEEE 16th, pp. 1–4. IEEE, Aydin (2008)
30. Kala, R., Vazirani, H., Shukla, A., Tiwari, R.: Offline handwriting recognition using genetic algorithm. International Journal of Computer Science **7**(1), 16–25 (2010)
31. Khedher, M.Z., Abandah, G.A., Al-Khawaldeh, A.M.: Optimizing feature selection for recognizing handwritten arabic characters. In: WEC (2), pp. 81–84. (2005)
32. Kodaz, H., Özşen, S., Arslan, A., Güneş, S.: Medical application of information gain based artificial immune recognition system (airs): Diagnosis of thyroid disease. Expert Systems with Applications **36**(2), 3086–3092 (2009)
33. Kumar, A., Nair, S.B.: An artificial immune system based approach for english grammar checking. In: Artificial Immune Systems, pp. 348–357. Springer, Berlin Heidelberg (2007)
34. Lawgali, A.: A survey on arabic character recognition. International Journal of Signal Processing, Image Processing and Pattern Recognition **8**(2), 401–426 (2015)
35. Lawgali, A., Angelova, M., Bouridane, A.: Hacdb: Handwritten arabic characters database for automatic character recognition. In: Visual Information Processing (EUVIP), 2013 4th European Workshop on, pp. 255–259. IEEE, Paris (2013)
36. Liang, C., Peng, L., Hong, Y., Wang, J.: An english letter recognition algorithm based artificial immune. In: Advances in Neural Networks–ISNN 2009, pp. 371–379. Springer, Berlin Heidelberg (2009)
37. Mamatha, H., Srikanta, M.K., Amrutha, K., Anusha, P., Azeemunisa, R.: Artificial immune system based recognition of handwritten kannada numerals. In: Advanced Materials Research, vol. 433, pp. 900–906. Trans Tech Publications (2012)
38. Mathur, S., Aggarwal, V., Joshi, H., Ahlawat, A.: Offline handwriting recognition using genetic algorithm. Institute of Information Theories and Applications FOI ITHEA, (2008)
39. Mori, S., Suen, C.Y., Yamamoto, K.: Historical review of ocr research and development. Proceedings of the IEEE **80**(7), 1029–1058 (1992)
40. Mozaffari, S., Faez, K., Faradji, F., Ziaratban, M., Golzan, S.M.: A comprehensive isolated farsi/arabic character database for handwritten ocr research. In: Tenth International Workshop on Frontiers in Handwriting Recognition. Suvisoft, La Baule (2006)
41. Nebti, S., Boukerram, A.: Handwritten characters recognition based on nature-inspired computing and neuro-evolution. Applied intelligence **38**(2), 146–159 (2013)
42. Nemmour, H., Chibani, Y.: Artificial immune system for handwritten arabic word recognition. In: Innovative Computing Technology (INTECH), 2013 Third International Conference on, pp. 463–466. IEEE, London (2013)
43. Patwardhan, S.S., Deshmukh, R.: Offline handwriting recognition using genetic algorithm. International Journal of Innovation Sciences and research **4**(9), 441–444 (2015)
44. Pechwitz, M., Maddouri, S.S., Märgner, V., Ellouze, N., Amiri, H., et al.: Ifn/enit-database of handwritten arabic words. In: Proc. of CIFED, vol. 2, pp. 127–136. Citeseer, Hammamet (2002)
45. Pechwitz, M., Maergner, V., El Abed, H.: Comparison of two different feature sets for offline recognition of handwritten arabic words. In: Tenth International Workshop on Frontiers in Handwriting Recognition. Suvisoft, La Baule (2006)
46. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. The Journal of Machine Learning Research **12**, 2825–2830 (2011)
47. Polat, K., Güneş, S.: Automated identification of diseases related to lymph system from lymphography data using artificial immune recognition system with fuzzy resource allocation mechanism (fuzzy-airs). Biomedical Signal Processing and Control **1**(4), 253–260 (2006)
48. Polat, K., Güneş, S.: Prediction of hepatitis disease based on principal component analysis and artificial immune recognition system. Applied Mathematics and computation **189**(2), 1282–1291 (2007)
49. Polat, K., Güneş, S.: Principles component analysis, fuzzy weighting pre-processing and artificial immune recognition system based diagnostic system for diagnosis of lung cancer. Expert Systems with Applications **34**(1), 214–221 (2008)
50. Polat, K., Güneş, S., Yosunkaya, Ş.: Pairwise classifier approach to automated diagnosis of disorder degree of obstructive sleep apnea syndrome: combining of airs and one versus one (ovo-airs). In: Proceedings of the International MultiConference of Engineers and Computer Scientists, Hong Kong (2009)
51. Polat, K., Şahan, S., Güneş, S.: Automatic detection of heart disease using an artificial immune recognition system (airs) with fuzzy resource allocation mechanism and k-nn (nearest neighbour) based weighting preprocessing. Expert

Systems with Applications **32**(2), 625–631 (2007)
52. Pornpanomchai, C., Wongsawangtham, V., Jeungudomporn, S.: Thai handwritten character recognition by genetic algorithm (thcrga). International Journal of Engineering and Technology **3**(2), 148 (2011)
53. Precup, R.E., David, R.C., Petriu, E.M., Preitl, S., Radac, M.B.: Fuzzy logic-based adaptive gravitational search algorithm for optimal tuning of fuzzy-controlled servo systems. IET Control Theory & Applications **7**(1), 99–107 (2013)
54. Rashad, M., Amin, K., Hadhoud, M., Elkilani, W.: Arabic character recognition using statistical and geometric moment features. In: Electronics, Communications and Computers (JEC-ECC), 2012 Japan-Egypt Conference on, pp. 68–72. IEEE, Alexandria (2012)
55. Rosenfeld, A.: Digital picture processing. Academic Press, United States (1976)
56. Sahlol, A., Suen, C.: A novel method for the recognition of isolated handwritten arabic characters. ArXiv e-prints:1402.6650 p. 13 (2014)
57. Serdouk, Y., Nemmour, H., Chibani, Y.: An improved artificial immune recognition system for off-line handwritten signature verification. In: Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, pp. 196–200. IEEE, Sfax (2015)
58. Serdouk, Y., Nemmour, H., Chibani, Y.: New off-line handwritten signature verification method based on artificial immune recognition system. Expert Systems with Applications: An International Journal **51**(C), 186–194 (2016)
59. Shatnawi, M.: Off-line handwritten arabic character recognition: A survey. In: Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV), p. 52. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), Las Vegas, Nevada (2015)
60. Thammano, A., Pravesjit, S.: Recognition of archaic lanna handwritten manuscripts using a hybrid bio-inspired algorithm. Memetic Computing **7**(1), 3–17 (2015)
61. Timmis, J., Neal, M.: A resource limited artificial immune system for data analysis. Knowledge-Based Systems **14**(3), 121–130 (2001)
62. Tomin, N., Zhukov, A., Sidorov, D., Kurbatsky, V., Panasetsky, D., Spiryaev, V.: Random forest based model for preventing large-scale emergencies in power systems. International Journal of Artificial Intelligence **13**(1), 211–228 (2015)
63. Twycross, J., Aickelin, U.: Information fusion in the immune system. Information Fusion **11**(1), 35–44 (2010)
64. Valdez, F., Melin, P., Castillo, O.: An improved evolutionary method with fuzzy logic for combining particle swarm optimization and genetic algorithms. Applied Soft Computing **11**(2), 2625–2632 (2011)
65. Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (airs): An immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines **5**(3), 291–317 (2004)
66. Watkins, A.B.: Airs: A resource limited artificial immune classifier. Ph.D. thesis, Mississippi State University (2001)
67. Yang, H., Li, T., Hu, X., Wang, F., Zou, Y.: A survey of artificial immune system based intrusion detection. The Scientific World Journal **2014**, 11 (2014)
68. Yang, Y.: Application of artificial immune system in handwritten russian uppercase character recognition. In: Computer Science and Service System (CSSS), 2011 International Conference on, pp. 238–241. IEEE, Nanjing (2011)
69. Yang, Y.: Handwritten armenian character recognition based on discrete cosine transform and artificial immune system. In: Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International, vol. 2, pp. 14–16. IEEE, Chongqing (2011)
70. Yang, Y.: Handwritten icelandic character recognition based on artificial immune system. In: Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International, vol. 2, pp. 17–20. IEEE, Chongqing (2011)
71. Yang, Y.: Handwritten nepali character recognition based on wavelet packet transform and artificial immune system. In: Computer Science and Service System (CSSS), 2011 International Conference on, pp. 442–445. IEEE, Nanjing (2011)
72. Zhang, Q., Luo, M., Xue, Y., Tan, J.: Multi-class text categorization based on immune algorithm. In: Education Technology and Training, 2008. and 2008 International Workshop on Geoscience and Remote Sensing. ETT and GRS 2008. International Workshop on, vol. 1, pp. 749–752. IEEE, Shanghai (2008)
73. Zhang, T., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Communications of the ACM **27**(3), 236–239 (1984)
74. Zăvoianu, A.C., Bramerdorfer, G., Lughofer, E., Silber, S., Amrhein, W., Klement, E.P.: Hybridization of multi-objective evolutionary algorithms and artificial neural networks for optimizing the performance of electrical drives. Engineering Applications of Artificial Intelligence **26**(8), 1781–1794 (2013)