Pattern Anal Applic (2008) 11:353–363
DOI 10.1007/s10044-008-0106-1

THEORETICAL ADVANCES

Prototype reduction using an artificial immune model

Utpal Garain

Received: 9 January 2007 / Accepted: 16 November 2007 / Published online: 12 February 2008
© Springer-Verlag London Limited 2008

Abstract Artificial immune system (AIS)-based pattern classification approach is relatively new in the field of pattern recognition. The study explores the potentiality of this paradigm in the context of prototype selection task that is primarily effective in improving the classification performance of nearest-neighbor (NN) classifier and also partially in reducing its storage and computing time requirement. The clonal selection model of immunology has been incorporated to condense the original prototype set, and performance is verified by employing the proposed technique in a practical optical character recognition (OCR) system as well as for training and testing of a set of benchmark databases available in the public domain. The effect of control parameters is analyzed and the efficiency of the method is compared with another existing techniques often used for prototype selection. In the case of the OCR system, empirical study shows that the proposed approach exhibits very good generalization ability in generating a smaller prototype library from a larger one and at the same time giving a substantial improvement in the classification accuracy of the underlying NN classifier. The improvement in performance has been statistically verified. Consideration of both OCR data and public domain datasets demonstrate that the proposed method gives results better than or at least comparable to that of some existing techniques.

U. Garain (✉)
Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute, 203, B.T. Road, Kolkata, India
e-mail: utpal@isical.ac.in

Keywords Nearest neighbor classification · Prototype selection · Artificial immune system · Clonal selection algorithm · Statistical significance

1 Introduction

The nearest-neighbor (NN) classification [1] scheme is one of the most popular supervised classification methods in pattern recognition (PR) tasks. It provides a simple and intuitive method for solving a great variety of real-world applications. In general, it performs well but suffers from two major drawbacks. (i) Storage and computational requirements: storage of the entire prototype dataset (library) requires large space. Moreover, comparison of each target (test) pattern with every prototype in the stored library makes the method computationally less attractive. (ii) Sensitivity: the NN classification scheme is quite sensitive to noise objects and outlier samples. To overcome these drawbacks researchers have proposed a prototype selection scheme that is a process by which a smaller set of prototypes is selected and used for classification. The resultant set may contain either members of the original prototype library or new patterns formed by using the original patterns. Such a method reduces storage and computing time requirement and if designed properly, it usually provides some improvement in classification accuracy.

Research in this area started immediately following the original NN scheme [8] was proposed. Hart [2] proposed the condensed nearest-neighbor (CNN) algorithm that initially puts a single prototype in the condensed set and then the remaining prototypes are considered one by one. Inclusion of a prototype into the condensed set is decided by finding its NN in the new set. If their labels match then the prototype is ignored. Swonger [3] considered both

addition and deletion of original prototypes to and from the condensed set. Gates [4] proposed an iterative deletion of redundant prototypes to form the condensed set. Following these initial efforts, many other approaches or modifications of the original methods have been reported in the literature. To get an overall idea on the advancement in this field, one may consults several articles in [5–19]. These articles nicely present how the research has been advancing in this area and at the same time, why prototype reduction is still considered as a challenging research problem.

This article proposes a biologically inspired approach for prototype condensation. The method is based on the artificial immune system (AIS) [20] that in the recent past has emerged as a viable computational framework for several engineering problems. The method has already found its successful applications in several engineering problems including computer security, network intrusion detection, fraud detection, etc. [21]. However, applications in PR have been investigated very recently. The approaches so far used to design AIS can broadly be classified into three groups, namely, immune network models [22], negative selection algorithms [23], and clonal selection algorithms (CSA) [24]. Using some of these paradigms in both supervised [25–28] and unsupervised [29, 30] pattern analyses have been attempted and encouraging results have been reported. The present study endeavors to contribute toward this on-going research effort.

This article investigates the role of an AIS-based framework for prototype data reduction task. Incorporation of immunological metaphor for this task is motivated by the amazing adaptability and generalization ability of an immune system in encountering pathogens. During its lifetime a mammalian body is exposed to a thousand of external pathogens but the immune system is able to protect the body by fighting against these limitless varieties of enemies. However, to do so an immune system need not memorize each and every pathogen. Rather, out of a single encounter the system adapts itself in such a way that makes it able to provide rapid response to any subsequent attack by a class of pathogens similar to the one seen earlier. The central contribution of the present study is to employ this biological idea for designing of an efficient method for prototype selection and for this purpose the CSA of immunology is followed.

The clonal selection model (popularly known as clonal selection algorithm, CSA) was first proposed by Burnet [31] and then further developed by Jerne [32]. The algorithm states that a mammal initially possesses a relatively small number of antibodies. The successful binding of an antibody to an antigen triggers the antibody to produce a large number of copies of itself. In this way, a pre-existing antibody is effectively selected by the antigen, which stimulates the chosen antibody to produce a multitude of clones. A computational model of CSA is available in [24, 30], etc. with minor variations among the implementations.

Contribution of this correspondence includes a novel formulation of the prototype selection problem from the immunological viewpoint so that the capability of CSA is explored. A goal-directed evaluation strategy is formulated to demonstrate the anticipated data reduction capability of CSA. This demonstration at first considers one of the most popular PR problems, namely, optical character recognition (OCR) system. An OCR system [34] using a NN classification-based recognition engine is involved to achieve the goal-directed evaluation of the proposed method. The improvement in character recognition accuracy by employing the proposed system has been verified by statistical tests.

In addition, comparison of the present approach with two of the commonly used techniques [9, 37] is presented in this context. Performance evaluation then considers standard benchmark datasets available in public domain and checks the efficiency of the proposed method. In addition, performance on these benchmark datasets has been compared with that of a recently proposed method [19] for prototype reduction.

The rest of the paper is organized as follows. Section 2 presents a general overview of the immune system, its key components that are essential to the development of an artificial version of the system. The section then introduces the clonal selection principle, which might constitute one of the most important features of the immune response to an antigenic stimulus. Section 3 describes how clonal selection model has been used for prototype selection. An upper level architecture of the proposed method is presented and then individual components of the system are explained in algorithmic manner. Section 4 outlines the goal-directed evaluation scheme to judge the efficiency of the proposed approach. The evaluation scheme considers a number of benchmark data sets as well as a real task, namely, OCR. Section 5 reports experimental results highlighting the achievements of CSA for prototype selection task. Statistical tests are also presented to show the significance of the results obtained in this experiment. In addition, this section compares the performance of the CSA-based approach with another commonly used methods for prototype reduction. Section 6 provides some concluding remarks.

2 Overview of the immune system

Several aspects of natural immune systems have been productive sources of inspiration for research in AIS. The description that follows is not comprehensive and is based primarily on discussions presented by Castro and Zuben [24], Carter [25], Watkins [26], and Timmis [30].

The immune system is composed of a great variety of cells among which two kinds of lymphocytes play a major role in the immune system: T cells (so called because they develop to maturity in the thymus gland), and B cells, which originate in bone marrow. When a pathogen invades the body, special cells called antigen-presenting cells (APC) process the pathogens so that their relevant features, called antigens are available on the surfaces of the APCs. An individual T cell or B cell responds like pattern matcher—the closer the antigen on a presenting cell is to the pattern that a T cell or B cell recognizes, the stronger the affinity of that T cell or B cell for the antigen. Although the B cells are considered as the major the immune response mechanism that multiplies and mutates to adapt to an invader, it is only when a T cell and B cell respond together to an antigen that the B cell is able to begin cloning it and mutating to adjust to the current antigen. That is why T cells are sometimes called helper T cells.

2.1 Clonal selection algorithm

The *clonal selection principle*, or *theory*, is the algorithm used by the immune system to describe the basic features of an *immune response* to an antigenic stimulus. The model was first proposed by Burnet [31] and then further developed by Jerne [32]. It advocates the idea that only those cells that recognize the antigens proliferate, thus being selected against those that which do not. Clonal selection operates on both T cells and B cells. When the body is exposed to an antigen, some sub-population of B cells responds by producing antibodies. Each cell secretes only one kind of antibody, which is relatively specific for the antigen.

A B cell that is sufficiently stimulated by the presented antigen rapidly produces clones of itself. At the same time, it produces mutations at particular sites in its gene that enable the new cells to match the antigen more closely. There is a very rapid proliferation (known as hyper-mutation or proliferation-I) of immune cells. These cells undergo successive generations of cloning (these generations are produced through proliferations known as stage-II proliferations or simply proliferation-II) with an aim of producing better and better matches for the antigens of the invading pathogen.

In fact, an antigen stimulates the B cell to proliferate (divide) and mature into terminal (non-dividing) antibody secreting cells, called plasma cells. While plasma cells are the most active antibody secretors, initial B cells, which divide rapidly during hyper-mutation phase, also secrete antibody, albeit at a lower rate. While B cells secrete antibodies, T cells do not secrete antibodies, but play a central role in the regulation of the B cell response and ensure the cell-mediated immune responses.

B cells that are not stimulated properly because they do not match any antigens in the body eventually die. This implies a resource limitation technique followed in the body. Assuming that the number of B cells in the body is finite, generated B cells compete for resources. The most stimulated B cells consume resources and the remaining cells are removed from the system. This meta-dynamics of the immune system applies a certain amount of evolutionary pressure to ensure that only the fittest (to fight against an invading antigen) B cells remain in the system.

Lymphocytes, in addition to proliferating or differentiating into plasma cells, can differentiate into long-lived B *memory cells*. Therefore, when a body has successfully defended against a pathogen, a comparatively small number of memory cells remain in the body for very long period of time. Memory cells circulate through the blood, lymph (fluid that carries lymphatic cells and invading antigens) and tissues, probably not manufacturing antibodies [33], but these memory cells rapidly recognize antigens similar to those that originally caused the immune response that created the memory cells. Therefore, the body's response to a second invasion of the same pathogen or a very similar invader is much more rapid and powerful than to a never-before-seen pathogen.

3 Prototype selection using CSA

Let an NN classifier use a prototype set P in which p_i represents an individual member of this set: $P = \{p_1, p_2,\ldots,p_k\}$. Each p_i has two attributes: *class label*: $p_i{\cdot}c \in C = \{c_1,c_2,\ldots,c_n\}$ and *feature vector*: $p_i{\cdot}f$. The goal of the prototype selection process is to find a condensed set P' from the original set P such that $|P'| \ll |P|$. The condensed set P' may contain either members of P or new patterns formed by using the elements of P.

In the proposed method, P is considered as the set of antigens $AG = \{ag_1, ag_2,\ldots,ag_k\}$ and CSA is employed to obtain P' which is synonymous to the immune memory, $IM = \{m_1, m_2,\ldots,m_m\}$ where m_i is a memory cell having two attributes similar to those of an individual antigen. For any m_i, $m_i{\cdot}c \in C = \{1, 2,\ldots,n\}$ is the class information and $m_i{\cdot}f$ is the feature vector. The perfect metrics proposed in [35] has been considered for generating feature vectors for individual prototype (i.e., antigen). Three types of features, namely, projection profile, contour and stroke directions are considered. This produces a vector of 448 dimensions out of which first 192 dimensions vary from 0 to 31, next 192 values are in [0, 63] and last 64 values are in [0, 127]. All these values are next converted into binary values. Therefore, each character is represented as binary string of length 2,560 ($192 \times 5 + 192 \times 6 + 64 \times 7$).

Figure 1 outlines the intermediate stages of CSA that takes an antigen set as input and produces an immune memory as output. The immune memory is considered as the condensed set of prototypes that is used during classification by the NN classifier. The algorithm used in this study borrows the major concepts from [26, 30] and some ideas from [25]. The algorithm involves three stages, namely, initialization of immune memory, clone generation, and selection of clones to update the immune memory. These stages are briefly discussed below.

Initialization. This stage deals with choosing some antigens as initial memory cells to initialize the immune memory. In the present study, only one antigen from each class is randomly chosen to initialize the immune memory (IM). It is to be noted that the number of initial cells has certain effect on system's performance as illustrated in [26].

Clone generation. For a given antigen ag_k, its closest match (say, m_i) is, at first, chosen from the existing IM as follows:

$$\text{stim}(agk,\ mi) \geq \text{stim}(agk,\ mj), \quad \text{for all } j \neq i \text{ and } mj \cdot c = agi \cdot c \tag{1}$$

The stimulation function stim(·) is used to measure the response of a B cell to an antigen or to another B cell. For the present implementation of CSA, the stimulation function returns a value in [0, 1] and is inversely proportional to the Hamming distance[1] between the feature vectors of the argument elements. The function stim(·) return '1' for the minimum hamming distance (i.e., 0) and '0' for the maximum distance (i.e., 2,560 for this experimentation).

After a memory cell m_i (renamed as m_{match}) is selected for the current antigen ag_k, m_{match} goes through a proliferation process (Proliferation-I), known as *somatic hyper-mutation* that generates a number of clones of m_{match}. The exact number of clones is determined by three parameters, namely, (i) hyper-mutation rate, (ii) clonal rate and (iii) stim(ag_k, m_{match}). Note that the first two parameters are user-defined.

Each clone is produced through mutation (controlled by MUTATION_RATE, a user defined parameter) at selected sites of m_{match}'s feature vector. No clone is an exact copy of m_{match}. The algorithms for proliferation-I and the generation of mutated clones are outlined in Algorithms I and II, respectively. These algorithms are similar to the ones described in [26]. On completion of hyper-mutation, a stimulation value is computed for each element $b_i \in B$ as stim(b_j, a_k). Here b_i denotes an individual B cell clone and B represents the entire cloned population.

To minimize the computational cost in generating clones, a resource limitation policy [22] is incorporated. The algorithm is described in Algorithm III. The algorithm follows an ad hoc scheme where half of the resources are given to the clones having same class of the current antigen. The other half is equally divided among clones of

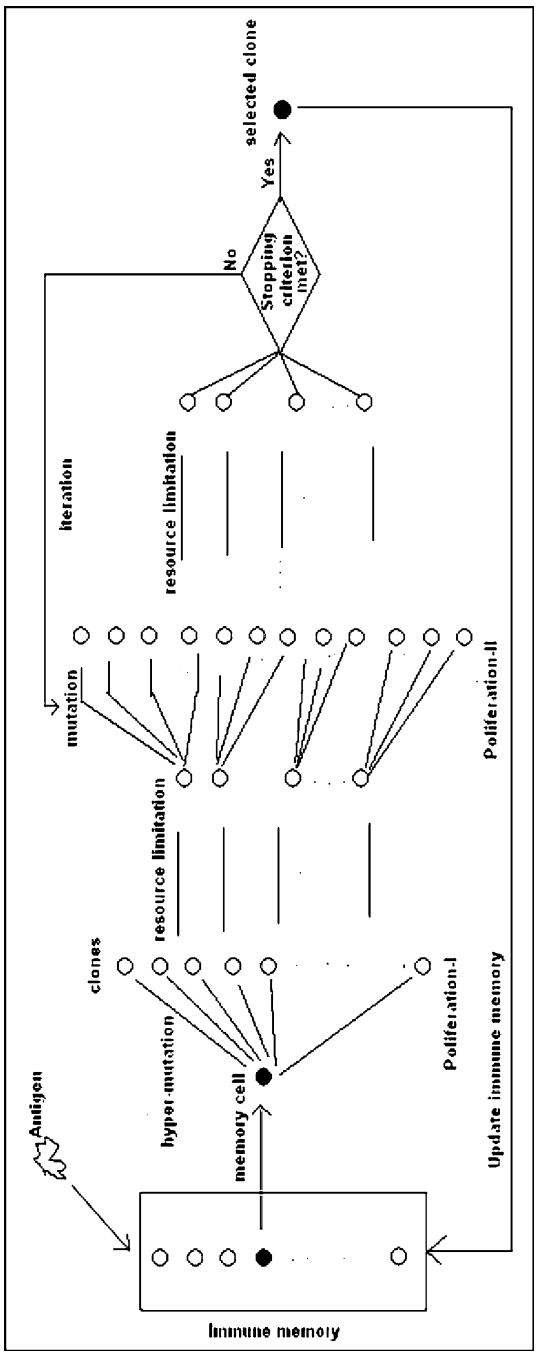

Fig. 1 A high-level architecture of the CSA used in the proposed approach

[1] Instead of Hamming distance, the present experiment also considers the use of Euclidean distance in measuring stimulation value. In this case, 448-dimensional features need not be converted into binary. Since the minimum and maximum values that can occur in each dimension are known, distance between a pair of patterns is normalized to give a stimulation measure in [0, 1]. However, by using Euclidean distance instead of Hamming distance no significant change was observed in the experimental results. All the results presented here are obtained when Hamming distance was used to measure stimulation.

other classes. In other words, the algorithm makes more resources available to the in-class (with respect to the current antigen) B cells than out-of- class cells.

Stopping criterion defined in Eq. 2 is used to terminate the iteration on an antigen ag_k. Say, B_1 is the collection of the in-class B cells, i.e. cells having class label $c = k$ and B_2 is the set of the rest of the out-of-class cells. The criterion in (2) then considers cells (in-class) in B_1 differently from the others, i.e. those (out-of-class) in B2. This gives a discriminative nature of the proposed algorithm making the learning criterion relevant to classification accuracy.

If this criterion in (2) is not met then further proliferation of existing (i.e., survived after resource limitation) B cells is invoked. In this stage (i.e., proliferation-II), each survived B cell, i.e., b_i is proliferated to produce a number of clones determined by the resources allocated to it. Proliferation-II process is similar to one for proliferation-I outlined in Algorithm I except the calculation of the number of clones to be generated from each surviving B cell (b_i). This number is determined only by the CLONAL_RATE and stim(ag_k, b_i) as used in Algorithm III to compute resources claimed by an individual B cell.

$$\frac{\sum_{i=1}^{|B_1|} b_i \cdot \text{stim}}{|B_1|} - \frac{\sum_{j=1}^{|B_2|} b_j \cdot \text{stim}}{|B_2|} > \text{STIMULATION\_THRESHOLD} \quad (2)$$

Clone selection and update of immune memory. Once the stopping criterion in Eq. 2 is met for an antigen, the most stimulated (w.r.t. the current antigen undergoing the proliferation stage) B cell among the survived ones is selected as a candidate (let $b_{\text{candidate}}$ denote this cell) to be inserted into immune memory. This process is outlined in Algorithm IV that is similar to one in [26]. This algorithm makes use of two parameters AS (average stimulation) and α (a scalar value). The parameter α is a user-defined one, whereas AS is measured from the input antigen set (i.e., the original prototype) as the average stimulation between all pairs of the mean values of the antigen classes.

Algorithm I. Hyper-mutation/proliferation-I

```
Let B is the set of B cell clones to be created after somatic hyper-mutation.
Initially B={m_match}.
Let N_c denote the number of clones and calculated as,
        N_c ← HYPER_MUTATION_RATE * CLONAL_RATE * stim(ag_k, m_i)
While (|B| ≤ N_c)
Do
    mut ← false   //mut is a Boolean variable
    Call mutate(m_match, mut)
    Let b_j denote a mutated clone of m_match
    If (mut) Then B ← B ∪ b_j
Done
```

Algorithm II. Production of mutated clones

```
mutate(x, flag){
For each element in x.f  // note that the feature vector x.f is basically a binary string of length 2560
Do
   Generate a random number, r in [0, 1]
   If (r < MUTATION_RATE) Then
        x.f ← toggle(x.f)
        flag ← true
   Endif
   If (flag)
     Generate a random number, r in [0, 1]
     Generate a random number, class in [1, n]
     If(r < MUTATION_RATE)
        x.c = class
Done
}
```

Algorithm III. Resource allocation

```
For each b_i ∈ B
        b_i.stim = stim(ag_k, b_i)
End for
Find minimum (minStim) and maximum (maxStim) from all b_i.stim values.

For each b_i ∈ B
        If (b_i.c == ag.c) b_i.stim = (b_i.stim -minStim)/(maxStim-minStim);
        Else b_i.stim = 1 - (b_i.stim -minStim)/(maxStim-minStim);
        b_i.resources = b_i.stim * CLONAL_RATE;
End for
class = 1
While (class ≤ n) // n is the total number of different classes
        resAllocated = ∑ b_i.resources, b_i.c = class
        If (class = ag.c)
                numResAllowed = (totalNumResources)/2
        Else    numResAllowed = (totalNumResources)/2*(n-1)
        While (resAllocated > numResAllowed)
                numResRemove = resAllocated – numResAllowed
                Find b_remove having the lowest stimulation among all b_i's s.t. b_i.c= class
                If (b_remove.resources ≤ numResRemove)
                        Remove bremove from B
                        resAllocated = resAllocated – b_remove.resources
                Else    b_remove.resources = b_remove.resources – numResRemove
        End while
        Class = class + 1
End while
```

Algorithm IV. Update of immune memory

```
CandStim ← stim(ag_i, b_candidate)
MatchStim ← stim (ag_i, m_match)
CellAff ← stim(m_match, b_candidate)
If (CandStim > MatchStim)
        IM ← IM ∪ b_candidate          // insertion into the immune memory
        If (CellAff > α × AS)
                IM ← IM – m_match      // memory replacement
```

4 Performance evaluation

The main objective of prototype selection is to improve the efficiency of the NN classifier in terms of storage and

computing time requirement, and the recognition accuracy. The amount of reduction in prototype data has direct reflection on the amount of improvement in storage and computing time but this may not guarantee the improvement in classification accuracy. Therefore, to conduct an objective evaluation, one needs to investigate how well the NN-classifier will perform the ultimate classification task. The proposed evaluation of prototype selection methods is in the context of character recognition, so the performance of the character recognizer is defined as the objective measure.

A practical OCR system is involved for this purpose. The report in [34] describes the development of an Indian language OCR (ILOCR) system, which was later productized as the first ILOCR package for commercialization. The system is a bi-lingual one in a sense that it can recognize (in either manner) two most popular Indic scripts, namely, Devanagari (Hindi) and Bengali (Bangla). The recognition engine implements an NN-classification scheme to classify more than 400 different shapes in both the scripts. Two different prototype libraries were used one for each of the two scripts. The features and distance measure used for classification are same for both the scripts and described in details in [36].

The initial prototype library used by the recognition engine was not developed by following a very judicious method. Prototypes were added to increase accuracy for a particular font face and then to tackle variations in font faces and styles other prototypes were added on need basis. This finally generates a large prototype library affecting the average recognition performance of the system. Although as far as computing time is considered the performance is quite acceptable (about 45 characters per second on a machine with average configuration; details can be found in [34]). Degradation in classification performance therefore calls for a prototype selection phase. The proposed method is then employed and its performance is evaluated against the performance of the original OCR system. The improvement in performance is then statistically verified. We call this as approach goal-directed evaluation, and it can be used to evaluate other prototype selection methods as well.

A comparative study makes use of this evaluation scheme to compare the proposed method with two of the important existing techniques for prototype selection. At first, a modified version of the original Hart's CNN method [2] is considered. The modification (named as MCNN) is proposed by Devi and Murty [9]. In the modified approach, the set of prototypes is built in an incremental manner. The process starts with a basic set of prototypes comprising one pattern from each class. The training set is classified using these prototypes. On the basis of the misclassified samples, a representative prototype for each class is determined and added to the set of basic prototypes. Prototypes that do not participate in the recent classification process are deleted. The training set is then classified again with the augmented set of prototypes. Representative prototypes for each class are again determined on the basis of the misclassified samples. This process of addition and deletion is repeated till all patterns in the training set are classified correctly.

Choice of the MCNN for comparing the efficiency of the present method is somewhat intentional, as MCNN has been experimentally verified to perform better than some other prototype selection methods that are designed based on genetic algorithm (GA), simulated annealing (SA), tabu search (TS), etc. The experiment in [9] considers different datasets to show the superiority of MCNN over CNN, GA, SA, and TS-based techniques. Success of MCNN for efficient prototype selection has motivated us to consider it for comparing performance of the present approach with MCNN.

Next, another popular prototype learning algorithm, namely self-organizing map (SOM) [37] is considered. The source code for SOM was downloaded from http://www.cis.hut.fi/research/som_lvq_pak.shtml and installed under windows platform. The comparison between the proposed CSA-based approach and SOM is made by first executing CSA and then number of prototypes in the reduced set is noted. This number is used to set the x- and y-dimensions of the map while applying SOM. A rectangular lattice is chosen as the topology of the map. With this experimental set-up, SOM is executed. The prototypes (called nodes in the rectangular grid) returned by the SOM form the reduced set of prototypes. Note that SOM is configured to return no. of nodes/prototypes nearly similar to the number obtained from CSA-based algorithm. Recognition performance using this set of prototypes selected by SOM dictates its ability of choosing right set of reference patterns.

4.1 Performance evaluation using benchmark datasets

The second part of performance evaluation scheme considers datasets that are publicly available. For this purpose, UCI repository of machine learning databases [38] and the Statlog Project [39] are consulted. Eight datasets as shown in Table 3 are selected for evaluating the performance of the proposed CSA-based approach. These datasets mostly involve both numeric and categorical features that are first converted into binary features. Therefore, the algorithms as presented in Sect. 3 are applied without doing any change. Note that they assume binary feature vectors.

Choice of these datasets is somewhat intentional, as many prototype reduction methods have already used these datasets to report their performance. For example, a very

recent study [19] has used several datasets from [38] and [39], reported their results, and compared with two other commonly used methods. This study [19] abbreviated as LPD (learning prototypes and distances) starts with an initial selection of a small number of randomly selected prototypes from the training set. Then it iteratively adjusts both the position (features) of the prototypes themselves and the corresponding local-metric weights, so that the resulting combination of prototypes and metric minimizes a suitable estimation of the probability of classification error. Gradient descent is used to solve the minimization problem to derive the adjustment rules.

As LPD is well tested with many datasets and found to be producing good results, in the present experiment, it has been selected as a reference study to compare our proposed CSA-based method. Following section presents the details of the CSA performance on both the OCR problem and the public domain datasets. It also compares these results with those obtained by some other methods.

5 Experimental results

The ILOCR described in [34] is used for the first phase of evaluation of the proposed method. The OCR deals with two major Indian scripts, namely, Devanagari (Hindi) and Bengali (Bangla). Our experiment considers both the scripts and nearly similar observations are noted. Therefore, instead of reporting results on both the scripts outcomes on Hindi dataset are presented here.

The Hindi dataset contains about 75,000 character samples that were ground-truthed to assist the design of the OCR. Character samples were collected from 50 document pages selected from five books printed in four different font faces. The NN classifier used by the OCR engine deals with 427 distinct classes. The number of samples for each character class in the dataset varies from one class to another. This number varies from a minimum of 37 to a maximum of 262. The initial prototype library is constructed following an incremental manner. The new prototypes were added to maximize the recognition accuracy on a document page in hand without considering its effects on recognition of other pages. This process generates prototype library of 12,823 patterns. In this library, number of prototypes for each character class varies from a minimum of 6 to 43.

Experiment shows that although the effort was to maximize the recognition accuracy of individual pages but the average accuracy comes down to an alarming level. For few pages high recognition accuracy such as 98–99% (in such cases errors are mostly attributed to segmentation method and therefore further addition of prototypes does not help) were obtained when they were considered in isolation but the average accuracy computed on all the pages becomes quite low as 91.2%. Analysis of this degradation in accuracy shows that many prototypes are used for correct classification of one or few symbols but they largely contribute to misclassification of many other symbols. The CSA-based prototype selection method described in Sect. 3 is employed at this stage to improve the situation.

Since performance of the proposed method depends on some parameters, the effects of parameters are first studied for two different measures: (i) size of the immune memory, i.e., size of the condensed prototype set and (ii) recognition accuracy. Results on Hindi dataset are shown in Fig. 2. After studying behavior of individual parameters, experimental set-up does fix the parameter values as follows: stimulation threshold = 0.89, number of resources = 400,

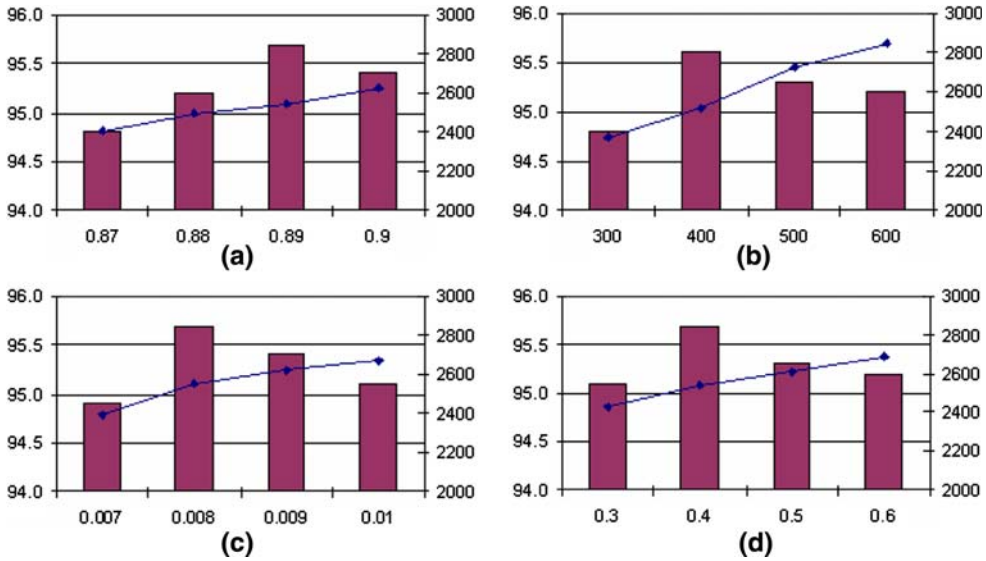

Fig. 2 Effect of different parameters on size of the condensed prototype set and recognition accuracy: **a** stimulation threshold (refer Eq. 2), **b** number of resources used for resource limitation, **c** mutation rate (refer Algorithm II), and **d** affinity threshold scalar, α as used in Algorithm IV

mutation rate = 0.008, affinity threshold scalar, $\alpha = 0.4$, hyper-mutation rate = 2 and clonal rate = 10 (the last two parameters are used in Algorithm I of Sect. 2).

With this parameter set-up, CSA-based prototype selection method generates a prototype library of 2,544 patterns. This gives more than 80% condensation of the original prototype data, thereby reducing the storage and computing time requirement by a large margin. The number of samples for each of the 427 classes varies from 3 to 15 in the new prototype set. An accuracy of 95.7% is achieved when recognition of the whole dataset is attempted using this condensed prototype library. This improves the recognition efficiency by more than 4%. Table 1 outlines the improvement in efficiency of the original NN-classifier achieved by the proposed prototype selection method.

5.1 Significance testing

The results reported in Table 1 are then verified statistically. The significant test consists a typical null hypothesis is that there is no difference in the two systems corresponding to the two rows in Table 1. Let system A and B denotes these two systems. For a test set consisting of samples for 427 classes, system A (old system) had an average recognition score of 0.912 and system B (proposed system) had an average recognition accuracy of 0.957. As measured by average recognition accuracy, system B performed 4.7% better than A, but is this a statistically significant improvement? To answer this question, significance of the results is tested statistically.

Each system produces a score for each class and on a per-class basis matched pairs of scores are obtained (since there are more than one sample for a class, all samples with the same class label are considered for computing per-class recognition score). Significance in light of this paired design is then evaluated. Two significance tests, namely, (i) randomization test and (ii) Student's paired *t* test are followed in the present experiment. Details of many significance tests including the two that have been used here can be found in [40]. Each method has its own criterion and null hypothesis. While there are fundamental differences in the null hypotheses, all the three tests aim to measure the probability (also known as *P* value) that the experimental results would have occurred by chance if systems A and B were actually the same system.

Table 1 Performance improvement through prototype selection

	#Prototypes	Storage requirement (MB)	Required milliseconds to classify 100 patterns	Classification accuracy (%)
Original system [24]	12,823	3.3	2,200	91.2
Proposed	2,544	0.7	550	95.7

5.1.1 *Randomization test*

For Fisher's randomization test [40], the null hypothesis is that systems A and B are identical and thus system A has no effect compared to system B on the average recognition accuracy for the given test samples.

Thus, if systems A and B are identical, then the decision to label one score for a test class as produced by system A or B is arbitrary. In fact, since there are 427 classes, there are 2^{427} ways to label the results under the null hypothesis. One of these labeling is exactly the labeling of the example that produced average accuracy of 0.912 for system A and 0.957 for system B.

Under the null hypothesis, any permutation of the labels is an equally likely output. We can measure the difference between A and B for each permutation. Since generating 2^{427} permutations are computationally difficult to manage, an alternative is to sample and produce a limited number of random permutations. The more samples, the more accurate will our estimate of *P* value be.

In the present experiment, we created 10,000 random permutations of systems A and B and measured the difference in average recognition accuracy for each arrangement. Table 1 shows difference in these two accuracies is 0.045. Of the 10,000 measured differences, 71 are $\leq$–0.045 and 68 are $\geq$0.045. This gives us a two-sided *P* value of (71 + 68)/10,000 = 0.0139. This shows that the difference of 0.045 is unlikely and thus we should reject the null hypothesis and report that system B has achieved a statistically significant improvement over system A.

5.1.2 *Student's paired* t *test*

The *t* test's null hypothesis is that systems A and B are random samples from the same normal distribution. To test this hypothesis, samples are paired as explained earlier. Differences between all pairs are then computed. Let X_D and s_D denote the average and standard deviation of those differences. The following equation is then used to compute *P* value: $\frac{\overline{X}_D}{s_D}\sqrt{N}$ where *N* is the number of paired samples (i.e., 427 in the present experiment). Following this experimental set-up, the two-sided *P* value of the *t* test is 0.0117, which is in agreement with the randomization (0.0139) test.

5.2 Comparison with MCNN and SOM

Performance of MCNN [9] is then investigated on this dataset. MCNN with delete operation is implemented. It generates a condensed library of 3,305 prototypes when executed on the original prototype set of 12,823 patterns. This achieves about 74% reduction in prototype data. When classification is attempted using this condensed set (denoted as $MCNN_1$ in Table 2), an accuracy of about 95.6% is observed, which is almost same as the accuracy obtained by CSA-based algorithm, although this is achieved using a larger set of prototypes (3,305 of MCNN vs. 2,544 of CSA).

However, if MCNN is restricted to generate nearly the same number of prototypes as given by CSA then it is found that the classification accuracy using MCNN-generated prototype set is somewhat lower than the one achieved by CSA. MCNN was forcefully stopped at the end of an intermediate iteration when it has already generated 2,543 (almost same as the number of prototypes given by CSA). Using this MCNN generated prototype library (denoted as $MCNN_2$ in Table 2), a classification accuracy of 94.8% is obtained which is somewhat lower than 95.7% given by CSA. This shows CSA is able to provide better representative prototypes than MCNN. Table 2 summarizes the comparative results between MCNN and the proposed approach.

As far as computing time is concerned, MCNN takes much longer time to generate the condensed prototype library. The number of iterations required by MCNN mostly dictates the time requirement of the algorithm. Note that each iteration involves the entire original prototype set to determine the misclassified patterns. Another time constraint originates from the fact that the condensed prototype set under construction is consulted in every iteration to identify members to be deleted. Finding out the representative patterns for each class in every iteration results in the third term of the time complexity.

Let N denote the number of patterns in original prototype set, n denote the number of members in condensed set, m denote the number of iterations required, and C denote the number of class labels. Therefore, the time complexity of MCNN can be represented as $\Theta(mN) + \Theta(n) + \Theta(C)$ from which it is clear that the first term dictates the overall time requirement. In the present experiment, MCNN takes 518 iterations ($m = 518$) to produce the condensed set of 3,305 patterns ($n = 3{,}305$) for 427 classes ($C = 427$).

On the other hand, time complexity of CSA-based approach is determined by two factors: (i) finding the memory element for proliferation and (ii) the number of iterations (say, k) required in the proliferation stage to find a properly stimulated clone. If n' denotes the size of the immune memory (i.e., size of the condensed prototype set) then the time complexity can be represented by $\Theta(n') + \Theta(kN)$, where N is the number of antigens (i.e., prototypes in the original set). Here also the second term, i.e., $\Theta(kN)$ dictates the overall time complexity. Experimentally it is verified that the average number of iterations (maximum and minimum are being 21 and 0[2]) required to produce properly mutated clone is 8 (i.e., $k = 8$). Therefore, as $\Theta(mN) \gg \Theta(kN)$ [$m = 518$ vs. $k = 8$] the overall time requirement of MCNN is quite higher than the proposed CSA-based approach.

Recognition accuracy obtained by CSA-based method is next compared with that of Kohonen SOM [36]. As mentioned earlier that the size of the rectangular grid of used in SOM is set by noting the number of prototypes retained by the CSA. Since CSA gives a reduced set of 2,544 patterns, a 51×50 grid of nodes is assumed while executing SOM. The reference vectors returned by SOM are then used for recognition of the test patterns. An accuracy of 95.4% is achieved. The comparison among the classification accuracies obtained by MNCC [9], SOM [36], and the proposed method is given in Table 2. This shows that the reduced prototype set given by the CSA-based method shows a classification power comparable with those achieved by other two existing method.

Table 2 Comparison between MCNN [9], SOM [36] and CSA (proposed)

	$MCNN_1$	$MCNN_2$	SOM	Proposed
#Prototypes in condensed set	3,305	2,543	2,550	2,544
Classification accuracy (%)	95.6	94.8	95.3	95.7

5.3 Benchmark dataset

As mentioned in the previous section, eight datasets are taken from publicly available repository [38] and [39]. The proposed method is applied to these dataset to get a set reference patterns to be used for classification. Since the number of patterns in the datasets used here is small (varies from 625 to 6,435), fivefold cross-validation technique has been applied to obtain the classification results. Each dataset is divided into five blocks using four blocks as a training set, and the remaining block as a test set. Reduced set of prototypes is formed from the training set and used for classifying the fifth block. Five runs are conducted so that each block is used as a test set. Table 3 reports the

[2] No iteration is needed if an antigen finds an exact match in the memory. In such a case, producing clones won't help to find any better B cell and that is why hyper-mutation phase is not invoked at all.

Table 3 Comparison between LPD and CSA (proposed)

Dataset	#Patterns	#Prototypes in reduced set		Classification accuracy (%)		
		LPD [19]	CSA	LPD [19] (best)	CSA	
					Best	Average
Australian	690	27	56	86.1	87.3	86.5
Balance	625	26	41	83.7	81.0	79.3
Cancer	685	28	48	96.6	97.7	97.1
Diabetes	768	31	57	74.0	71.3	70.3
DNA	3,186	128	192	95.1	96.3	95.0
German	1,000	40	67	74.0	75.5	74.6
Satimage	6,435	256	415	89.4	91.1	90.5
Vehicle	846	35	58	72.6	74.5	72.8

results averaged over these five runs as well as the best ones.

This experiment is conducted following the set-up proposed in a recent study [19]. This helps to make a comparison of the LPD approach proposed in [19] and the current one. LPD is found to perform better/comparable with two other commonly used methods, namely, L_2 metric and class-dependent Mahalanobis (CDM) distance. In the present experiment, when CSA results are compared with that of LPD, it is observed LPD shows more data condensation capability than that of CSA. LPD gives about 95% condensation while CSA gives about 93–90%. Therefore, reference patterns retained in the reduced prototype set is more in CSA than that of LPD.

When classification accuracies are compared, CSA gives accuracies comparable to or better than that of LPD. For some datasets such as Australian, Cancer, German, Satimage, Vehicle the average case accuracies are slightly better than the best-case accuracies of LPD. However, for many cases this improvement is found to be due to the presence of more number of prototypes in the reference library.

6 Conclusion

This paper presents a biologically inspired method for prototype selection which is primarily effective in improving the classification performance of NN classifier and also partially in reducing its storage and computing time requirement. An immune model, namely, CSA-based approach is implemented for the proposed prototype selection task. A practical PR system and datasets from publicly available repositories are considered to evaluate the performance of the method. Improvement of the classification accuracy is demonstrated. Comparison of the present method with another popular prototype selection techniques clearly brings out the potentiality of the proposed immune-based paradigm.

The CSA presented here is one of the simplest versions among the modifications proposed in recent times. Such a choice was intentional to investigate the initial performance of a CSA-based approach for prototype selection method. The algorithm is one-pass, i.e., each member in the original prototype library is consulted once. Multi-pass approach [28] may results in better performance. The future extension of this on-going study will consider such modifications in the CSA to achieve better efficiency.

References

1. Cover TM, Hart PE (1967) Nearest neighbor pattern classification. IEEE Trans Inform Theory 13:21–27
2. Hart PE (1968) The condensed nearest neighbor rule. IEEE Trans Inform Theory (IT) 14(3):515–516
3. Swonger CW (1972) Sample set condensation for a condensed NN decision rule for pattern recognition. In: Watanab S (ed) Frontiers of pattern recognition. Academic Press, New York, pp 511–519
4. Gates GW (1972) The reduced nearest neighbour rule. IEEE Trans Inform Theory 18(3):431–433
5. Sanchez JS, Pla F, Ferri FJ (1995) Prototype selection for the nearest neighbour rule through proximity graphs. Pattern Recognit Lett (PRL) 18(6):507–513
6. Skalak DB (1995) Prototype selection for composite nearest neighbor classifiers. PhD thesis, Computer Science, University of Massachusetts Amherst, USA
7. Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38(3):257–286
8. Brighton H, Mellish C (2002) Advances in instance selection for instance-based learning algorithms. Data Min Knowl Discov 6:153–172
9. Susheela Devi V, Narasimha Murty M (2002) An incremental prototype set building technique. Pattern Recognit 35:505–513
10. Mollineda R, Ferri FJ, Vidal E (2002) An efficient prototype merging strategy for the condensed 1-NN rule through class-conditional hierarchical clustering. Pattern Recognit 35:2771–2782
11. Pekalska E, Duin RPW (2002) Prototype selection for finding efficient representations of dissimilarity data. In: Sixteenth international conference on pattern recognition (ICPR), vol 3, pp 37–40
12. Sanchez JS, Barandela R, Marques AI, Alejo R, Badenas J (2003) Analysis of new techniques to obtain quality training sets. Pattern Recognit Lett (PRL) 24(7):1015–1022

13. Cano JR, Herrera F, Lozano M (2003) Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. IEEE Trans Evol Comput 7(6):561–575
14. Sanchez JS (2004) High training set size reduction by space partitioning and prototype abstraction. Pattern Recognit 37(7):1561–1564
15. Li Y, Hu Z, Cai Y, Zhang W (2005) Support vector based prototype selection method for nearest neighbor rules. Advances in natural computation. Lecture notes in computer science, vol 3610. Springer, Berlin, pp 528–535
16. Barandela R, Ferri FJ, Sanchez JS (2005) Decision boundary preserving prototype selection for nearest neighbor classification. Int J Pattern Recognit Artif Intell (IJPRAI) 19(6):787–806
17. Huang DD, Chow TWS (2006) Enhancing density-based data reduction using entropy. Neural Comput 18(2):470–495
18. Paredes R, Vidal E (2006) Learning prototypes and distances: a prototype reduction technique based on nearest neighbor error minimization. Pattern Recognit 39(2):180–188
19. Kim S-W, John B Oommen (2003) A brief taxonomy and ranking of creative prototype reduction schemes. Pattern Anal Appl (PAA) 6(3):232–244
20. Dasgupta D, Ji Z, Gonzalez FF (2003) Artificial immune system (AIS) research in the last five years. In: Congress on evolutionary computation (CEC'03) 1:123–130
21. Dasgupta D (1998) An overview of artificial immune systems and their applications. In: Dasgupta D (ed) Artificial immune systems and their applications. Springer, Berlin, pp 3–21
22. Zheng Tang, Koichi Tashima, Cao QP (2003) Pattern recognition system using a clonal selection-based immune network. Syst Comput Japan 34(12):56–63
23. Ji Z, Dasgupta D (2004) Real-valued negative selection algorithm with variable-sized detectors. In: Proceedings of GECCO. LNCS, vol 3102, pp 287–298
24. de Castro LN, Zuben FVJ (2002) Learning and optimization using the clonal selection principle. IEEE Trans Evol Comput Spec Issue Artif Immune Syst 6:239–251
25. Carter HJ (2000) The immune system as a model for pattern recognition and classification. J Am Med Inf Assoc 7(3):28–41
26. Watkins AB (2001) AIRS: a resource limited artificial immune classifier. Master's dissertation, Department of Computer Science, Mississippi State University
27. Garain U, Chakraborty PM, Dutta Majumder D (2006) Improvement of OCR accuracy by similar character pair discrimination: an approach based on artificial immune system. In: Proceedings of the 18th international conference on pattern recognition (ICPR), August 2006, Hong Kong II, pp 1046–1049
28. Garain U, Chakraborty PM, Dasgupta D (2006) Recognition of handwritten indic script using clonal selection algorithm. In: Bersini H, Carneiro J (eds) 5th international conference on artificial immune systems (ICARIS), 2006, LNCS, vol 4163. Springer, Berlin, pp 256–266
29. de Castro LN, Timmis J (2002) Artificial immune systems: a novel approach to pattern recognition. In: Alonso L, Corchado J, Fyfe C (eds) Artificial neural networks in pattern recognition. University of Paisley, pp 67–84
30. Timmis J (2001) Artificial immune systems: a novel data analysis techniques inspired by the immune network theory. Ph.D. thesis, University of Wales, Aberystwyth
31. Burnet FM (1959) The clonal selection theory of acquired immunity. Vanderbuilt University, Nashville, TN, USA
32. Jerne NK (1974) Towards a network theory of the immune system. Ann Immunol (Inst Pasteur) 125C:373–389
33. Perelsen AS, Oster GF (1979) Theoretical studies of clonal selection: minimal antibody repertoire size and reliability of self-nonself discrimination. J Theor Biol 81:645–670
34. Chaudhuri BB, Garain U, Mitra M (2003) On OCR of the most popular Indian scripts: Devnagari and Bangla," Technical report, no. TR/ISI/CVPR/03/2003, Indian Statistical Institute, Kolkata, August 2003. A product named Chitrankan is developed based on this research (http://www.cdac.in/HTML/gist/products/chitra.asp)
35. Baird HS (1993) Perfect metrics. In: Proceedings of the second international conference on document analysis and recognition, Tsukuba, Japan, pp 593–597
36. Garain U, Chaudhuri BB (1998) Compound character recognition by run number based metric distance. In: Proceedings of the IS&T/SPIE's 10th international symposium on electronic imaging: Science & Technology, SPIE, vol 3305. San Jose, CA, USA, pp 90–97
37. Kohonen T (1990) The Self-organizing map. Proc IEEE 78(9):464–1480
38. Blake C, Keogh E, Merz C. UCI repository of machine learning databases. http://www.ics.uci.edu/mlearn/MLRepository.html
39. D. Statistics and M.S.S.S. University, Statlog Corp. http://ftp.strath.ac.uk
40. Box GEP, Hunter GW, Hunter SJ (1978) Statistics for experimenters. Wiley, New York

Author Biography

Utpal Garain received his bachelor and master degrees in computer science and engineering in 1994 and 1997, respectively from Jadavpur University, India, and Ph.D. degree from Indian Statistical Institute in 2005. Dr. Garain started his career in software industry and later on, he joined Indian Statistical Institute where he is, at present, serving as an Assistant Professor. His research interest includes document image analysis including document compression and OCRs, technology development in Indian languages, natural-language processing, artificial immune systems, etc. Dr. Garain has been invited to serve in the program committees of several international conferences. Moreover, he has been regularly reviewing papers for several international journals of high repute. For his significant contribution in pattern recognition and its applications for language engineering, Dr. Garain received prestigious Young Engineer Award in 2006 from the Indian National Academy of Engineering (INAE).