A New Classifier Based on Resource Limited Artificial Immune Systems

Andrew Watkins
Computing Laboratory
University of Kent at Canterbury
and MPI Software Technology, Inc.
abw5@ukc.ac.uk

Lois Boggess
Intelligent Systems Laboratory
Department of Computer Science
Mississippi State University
lboggess@cs.msstate.edu

Abstract: This paper presents a new tool for supervised learning, modeled on resource limited Artificial Immune Systems. A supervised learning system, it is self-regulatory, efficient, and stable under a wide range of user-set parameters. Its performance is comparable to well-established classifiers on a variety of testbeds, including the iris data, the diabetes classification problem, the ionosphere problem, and the rock/metal classification problem for mine detection.

I. INTRODUCTION

Artificial Immune Systems are a relatively new biologically-motivated paradigm which has been explored for less than a decade. Yet they have been applied to a wide variety of tasks, including recognition tasks, such as intrusion detection [1] [2], and clustering tasks (see for example the work of Timmis and others [3], [4]). Success in these tasks suggests that the paradigm could lead naturally to a model for classification and supervised learning. Such a model has been surprisingly elusive, however. We are aware of only one previous classification system based on Artificial Immune Systems, with modest success [5].

In this paper, we introduce a new classifier based on the paradigm of Resource-Limited Artificial Immune Systems. The classifier, dubbed AIRS (Artificial Immune Recognition System), shows performance comparable to a number of widely used and well-respected classifiers on a suite of classic problems: the iris data, the diabetes classification problem, the ionosphere problem, and the rock vs. metal recognition problem for detection of mines.

Moreover, AIRS has a number of attractive features as a classification system [6]:

- It is self-regulatory in the sense that a user need not guess an appropriate architecture for the system. AIRS conforms its resources to the data presented to it.
- It is capable of generalization.
- After training, the resulting system which performs classification on real-world data may be significantly smaller than the training data set.
- It has a number of user-adjustable parameters which can be used to fine-tune it to the needs of a given problem.
- Its performance is stable over a wide range of settings of the adjustable parameters. Tweaking the parameters can result in several percentage points' difference in performance, but the system is remarkably consistent over the majority of settings.

In this paper we discuss the principles on which the AIRS classifier is built. We present a synopsis of the algorithm underlying the classifier. We demonstrate its behavior under a variety of starting conditions, and we present its performance on a suite of benchmark classification problems.

II. IMMUNE SYSTEMS

There are many aspects of natural immune systems that may serve as metaphors for computational systems. Any specific artificial immune system is generally based on a subset of the available metaphors, and different artificial immune systems select different aspects of the biological immune systems for motivation. We present here several aspects of natural immune systems which have been productive sources of inspiration for research in artificial immune systems which in turn have affected our own approach. The description that follows is not comprehensive and is based primarily on models presented by Carter [5] and Timmis et al. [3].

A biological immune system has two broad response systems. One is innate immunity, which is general rather than specific to invading pathogens and which is not affected in strength or specificity by exposure to new pathogens. This part of the immune system is not normally modeled by AIS systems. The other immune response in biological systems is based on two kinds of lymphocytes in the body: T-cells, so named because they originate in the thymus gland, and B-cells, which originate in bone marrow. When a pathogen invades the body, special cells called antigen presenting cells process the pathogens so that their relevant features, called antigens, are available on the surfaces of the antigen presenting cells. An individual T cell or B cell responds like a 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. T cells are sometimes called helper T cells because in nature, although the B cells are 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 itself and mutating to adjust to the current antigen.

Once a B cell is sufficiently stimulated though close affinity to a presented antigen, it rapidly produces clones of itself. At the same time, it produces mutations at particular sites in its gene which enable the new cells to match the antigen more closely. There is a very rapid proliferation of immune cells, successive generations of which are better and better matches for the antigens of the invading pathogen.

B cells which are not stimulated because they do not match any antigens in the body eventually die. However, when a body has successfully defended against a pathogen, a comparatively small number of memory cells remain in the body for very long periods of time. These memory cells recognize antigens similar to those that originally caused the immune response that created the memory cells, so that the body's response to a later invasion of the same pathogen or a very similar invader is much more rapid and powerful than to a never-before-seen invader.

A somewhat controversial theory of immune systems holds that B cells are interconnected in networks throughout the body, and that B cells are stimulated and suppressed by other B cells in their networks, as well as by their affinity to an antigen.

III. Artificial Immune systems

Artificial immune systems use various metaphors from natural immune systems. Some systems, for example, use only one kind of lymphocyte - only T cells, or only B cells. In artificial systems it is not unusual for the representation of an antigen to be the same as the representation of the B or T cell that recognizes it: both antigen and lymphocyte are represented as a feature vector in the same feature space.

In the early work of Timmis, Hunt, and Neal ([3], [4]), a clustering algorithm based on metaphors related to B cells behaves as follows: First, a randomly selected proper subset of the training data is taken as the initial population of B cells. The remaining vectors of the data are presented one at a time to the system. For each such "antigen", the existing B cells' affinities are calculated. Affinity is inversely proportional to Euclidean distance in the feature space. That is, in their system, all features are normalized so that the maximum distance between any two feature vectors is 1.0. Suppose a feature space consists of n dimensions. Let antigen a = <a_1, a_2, a_3, … a_n>, B cell b_i = <b_{i1}, b_{i2}, … b_{in}>, and B cell b_j = <b_{j1}, b_{j2}, … b_{jn}>. Then the distance between antigen a and B cell b_i is given by

$$dist_{ab_i} = \sqrt{\sum_{k=1}^{n}(a_k - b_{ik})^2} \, .$$

In early systems based on these principles, the number of B cells produced tended to swamp the system. More recently, Timmis and others experimented with resource limited Artificial Immune Systems[7]. In particular, they introduced the concept of an Artificial Recognition Ball (ARB), which has the same representation as a B cell, but may stand for any number of identical B cells. Each ARB represents a certain number of the B cells, or resources, and the total number of resources of the system is bounded.

We have adopted the concept of ARBs with resources, though in our view it is probably not important whether the resources are taken literally to be the number of B cells allowed to the overall system. What is important is that there be some resource (possibly abstract) which is limited and for which the ARBs must compete.

IV. Summary of concepts from other work relevant to the AIRS classifier

- ARBs. These are like B cells, but represent groups of identical B cells. The number of B cells is the number of "resources" represented by the ARB.
- The total number of resources in the system is limited.
- Antigens and ARB antibodies are feature vectors in the feature space.
- Affinity between the antibody of an ARB and an antigen is taken to be Euclidean distance in the feature space (although it could be some other measure).
- A stimulated ARB undergoes affinity maturation and begins to produce clones and mutated offspring. Identical clones are represented by an increase in resources for the corresponding ARB.
- Mutated offspring get a chance to compete for resources.
- Those ARBs which get new resources result in the deletion of resources from the weakest ARBs (where "weakest" means having the worst stimulation).
- The B cell network metaphor allows for the interaction of B cells with other B cells and ARBs with other ARBs.
- Feature vectors are normalized so that the largest distance is 1.0 and the smallest is 0.0.
- Memory cells have permanence, in contrast to most B cells.

V. The AIRS contributions:

- A mechanism for supervised learning and classification.
- While resources could be interpreted as representing the number of identical B cells, it is possible to interpret them simply as an abstract quantity which is limited in the overall system, for which competition takes place.
- Class now matters. B cells are rewarded for strong affinity toward antigens of the same class and weak affinity to antigens of different classes.
- Allocation of resources takes class into account.
- Mutation takes class into account.
- Memory cell replacement takes class into account.

VI. The algorithm

The AIRS algorithm has four stages. The first is performed once at the beginning of the process.

1) Normalization and initialization

The other stages constitute a loop and are performed for each antigen in the training set.

2) ARB generation
3) Competition for resources and nomination of candidate memory cell
4) Promotion of candidate memory cell into memory cell pool

Bear in mind that there are many ARBs, but in the final trained system, only the special subset of them constituting the memory cells will be used to classify test instances.

A. Initialization

This stage can be thought of as a preprocessing step.

1) Normalize all feature vectors so that distances between antigents and ARBs or between two ARBs is in the range [0,1].

2) Calculate the affinity threshold, which will be used to determine whether a new memory cell is close enough to an existing memory cell to replace it (see section D below). For training sets of fixed size, the affinity threshold is the average affinity, calculated pairwise over all training instances. If the training set is thought of as antigens, ag_i, then

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

where affinity is the Euclidean distance in the normalized feature space.

3) (optional) Seed the memory cell population and the ARB population with zero or more antigens in each population, randomly selected from the training data.

With our current version, training consists of a single pass through the training data. For each antigen *ag* in the training set, perform the following three steps.

B. ARB generation

1) Identify the memory cell which has the same class as the antigen and which is most stimulated by the antigen ag as follows:

$$mc_{match} = \begin{cases} ag & \text{if there are no memory cells having the same class as ag in the memory pool. Add ag to the pool.} \\ \underset{mc \in MC_{ag.class}}{\arg\max}\ stimulation(ag, mc) & \text{otherwise} \end{cases}$$

where stimulation is defined as 1 – $dist_{ag,mc}$ using the distance formula on the previous page (Euclidean distance).

2) Once the memory cell with highest stimulation is identified, it generates new ARBs. Let NumClones = hyperClonalRate * clonalRate * stimulation(ag,mc_{match}). The hyperClonalRate and clonalRate are integer values set by the user. The clonalRate is used to determine how many clones are produced by ARBs and memory cells. A typical value is 10. The hyperClonalRate is a multiplier which ensures that a hypermutating memory cell produces more new cells than a standard ARB. If hyperClonalRate is 2, then a hypermutating memory cell generates twice as many new cells as an ARB in step C.5 below. AIRS creates *NumClones* new clones of mc_{match}, where each feature of the clone can be mutated with probability *mutationRate*. (The variable *mutationRate* is a user-defined constant between zero and one.) The class of the clone is also subject to mutation at this same rate. All of the new clones are added to the ARB population, which until now has either been empty or has consisted of ARBs which were created in response to previous training instances.

C. Competition for resources and nomination of new memory cell

3) Now consider a set of ARBs which consists of

AB = mc_{match} + all the new clones + all ARBs left over from previous antigen reactions

These will compete for resources based on stimulation by the current antigen.

a) Find the maximum stimulation and minimum stimulation among all the ARBs in our set, regardless of class.

b) For each ab$\in$AB, normalize its stimulation according to the formula

$$ab.stim = \frac{ab.stim - minstim}{maxstim - minstim}$$ if ab is of the same class as the antigen ag

$$ab.stim = 1 - \frac{ab.stim - minstim}{maxstim - minstim}$$ if ab belongs to a different class than ag.

c) For each ab$\in$AB calculate ab's resources based on stimulation level:

ab.resources = ab.stim * clonalRate

d) Sum all resources. Allow half of all resources to be allocated to the ARBs belonging to the same class as the antigen. Split the remaining half of all resources evenly among the ARBs which belong to other classes. If the sum of resources just allocated to the ARBs for a class exceeds the allowance for that class, resources are removed from the least stimulated ARBs first. Any ARB whose resources are reduced to zero by this process is removed from the ARB pool.

4) A stopping criterion is calculated at this point. It is met if the average stimulation level for every class is above a threshold value set by the user.

5) Meanwhile, all surviving ARBs are allowed to generate mutated clones, although at a lower rate than the

hypermutating memory cell of step b).

6) If the stopping criterion has not been met, repeat, beginning at step c).

For any pass after the first through steps 4) to 6), if 4)'s criterion is met the process is stopped before step 5).

7) Nominate the ARB of the antigen's class that was most stimulated for inclusion in the memory cell pool.

D. Promotion of candidate memory cell to memory cell pool

If the new candidate for the memory cell pool, mc_{cand}, is a better fit for the presenting antigen than the best existing memory cell, mc_{match}, it will be added to the pool. Moreover, if the distance between mc_{cand} and mc_{match} is less than the affinity threshold times the affinity threshold scalar (a user adjustable parameter) then mc_{cand} actually replaces mc_{match} in the memory cell pool and mc_{mach} is discarded.

All of the above constitutes the training resulting from presentation of one antigen from the training set. Our initial version of AIRS performs only one pass through the training set.

The figures that follow show the performance of AIRS on a non-linearly separable two-dimensional 2-way classification task.

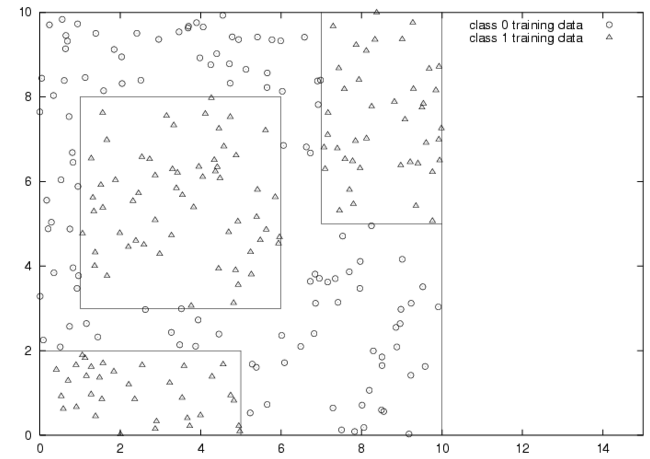

Figure 1. Training data of 250 antigens for a two-dimensional 2-class classification task [6].

As shown in Figure 1, class 0 is a complex shape, while class 1 consists of three regions in the plane. The training data were generated by a random number generator, and as can be seen are sparsely represented in some areas of the plane and somewhat crowded in other areas.

Figure 2 shows the memory cells which were generated from the training data superimposed on top of the training data. In the example shown, only three of the memory cells were originally antigens from the training set. All others are mutations of clones. These surviving mutations became permanent memory cells because they were successful generalizations of the features of the training antigens.

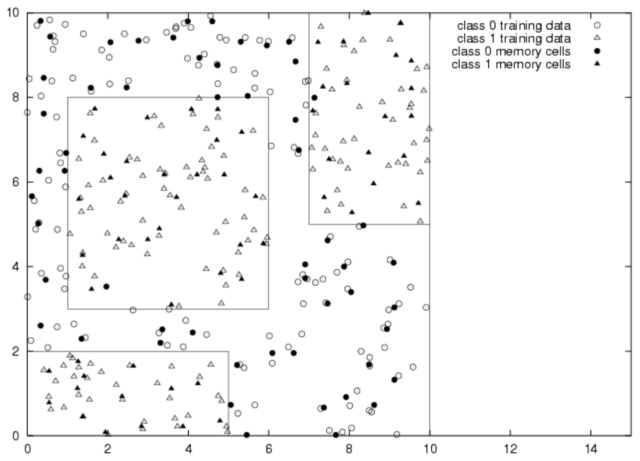

Figure 2. Training data with memory cells superimposed [6].

As can more clearly be seen in Figure 3, below, there are fewer memory cells than training instances.

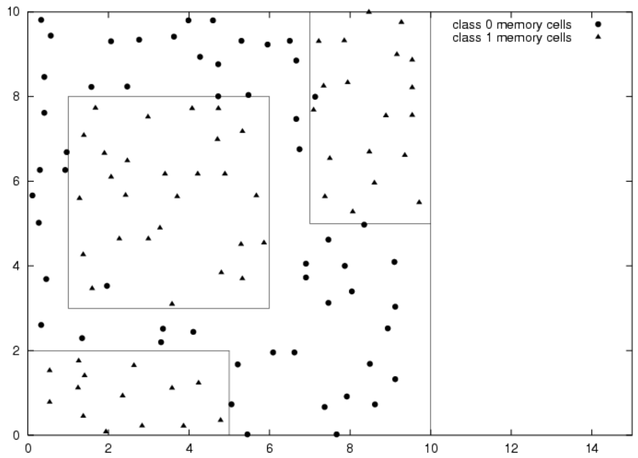

Figure 3. Memory cells generated during training on the 250 antigens of Figure 1 [6].

One fact that is also evident in Figure 3 is that for the training session that generated this set of memory cells, two memory cells are wrong. That is, two cells belonging to class zero are inside regions that belong to class one. We believe that these cells were produced during the earliest stages of training when there were few known data points for either class. None of the later successful memory cells were near enough to them to replace them, so they continued to survive. While it is in the nature of memory cells to have permanence unless replaced by a more successful and more general memory cell, we are considering relaxing this aspect of the model for the earliest memory cells generated during a training session.

Classification of test data is accomplished by majority vote of the k nearest memory cells to the presented test antigen. The value of k is set by the user. Experiments described in [6] use k values from 1 to 10 on a variety of classification tasks using readily available public data. For a detailed description of experiments using the data of the preceding figures, see [8].

VII. Performance on classification testbeds

AIRS has been tested on a variety of classification problems. In addition to being tested on the problem set alluded to in the previous figures, it has been used to classify four benchmark data sets taken from the repository of the University of California at Irvine[9]: the Fisher iris data set, the Pima diabetes data set, the Ionosphere data set, and the Sonar data set. As described in [6], the user-assignable parameters – number of seed cells, maximum number of system resources, stimulation threshold, mutation rate, affinity threshold scalar, and the k for k-Nearest-Neighbor were varied. For example, one set of parameters might be a single seed cell, 200 "resources", a stimulation threshold of 0.9, a mutation rate of 10%, an affinity threshold scalar of 0.2, and k value of 1. All results reported are averages over multiple runs. A typical experiment consisted of three runs of five-way cross-validation for a single setting of parameters. An exception was the Ionosphere data, where it is conventional to use the first 200 items in the data set as training and the remaining 151 instances as test. For this data set, cross-validation was not performed. However, three trials were performed for all parameter settings, and the resulting average was reported.

What was found was that the resulting classifiers are remarkably stable over a wide range of values for the user-settable parameters. Varying the number of seed cells from 1 to 100, for example, had an appreciable effect on classification accuracy on the training set, but clearly improved accuracy on the test set only for the sonar classification task. Varying system resources from 50 to 500 did not result in any clear trend: it seems likely that the important part of the algorithm is that there be competition for the resources, not that there is some particularly appropriate number of resources to be offered. Varying the stimulation threshold from 0.1 to 1.0 showed that classification of two of the data sets benefited slightly from higher thresholds. But accuracy *decreased* slightly on the iris data set, and on the ionosphere data, best results were found at low and high thresholds, and poorest for moderate thresholds (e.g., 0.4). Mutation rate was varied from 0.1 to 1.0. With the exception of the diabetes classification problem, which was virtually unaffected by variation in mutation rate, there was a general tendency for results to be better with modest mutation rates rather than high ones.

The one parameter which clearly did have an influence on classification performance was the activation threshold scalar. When varied between 0.1 and 1.0, accuracy dropped significantly for values above 0.5. For two of the four sets, accuracy dropped markedly when this parameter exceeded 0.3.

The best value for k in the k-nearest neighbor comparison with memory cells appears to be idiosyncratic to the problem: tests were run for k = 1 to k = 10. For two of the data sets, k=3 gave the best results. The diabetes classification problem showed slow improvement as k was increased throughout the range from 1 to 10. Since one of the ten best classifiers for this classification task is a k-nearest neighbor classifier with k=22 [10] (see table below), it is possible that AIRS would benefit from a higher k as well.

Duch [10], [11] publishes the results of applying a large number of classifiers against many of the benchmark classification problems. As with AIRS, most of the classifiers were run using cross-validation, and the data for the Ionosphere problem were treated in accordance with established practice. Shown on the next page is a table that encapsulates the results provided by Duch, inserting AIRS in the appropriate location in the ordered list for each classification problem. (Note that the order and values for the Ionosphere data have been replicated from Duch, but the order of that list suggests that the actual accuracy of the non-linear perceptron may have been 93.0 rather than 92.0)

As is evident, AIRS compares well with some of the best general purpose classifiers available. Indeed, on every benchmark problem it outperforms a number of well-respected classifiers. The only task for which its performance was not among the top 10 was the Diabetes set, which is a difficult problem for all classifiers. We have a number of refinements in mind for AIRS in any event, and we look forward to determining whether they improve its performance on this data set, among others.

VIII. Future work

There are two modifications to the basic AIRS algorithm that we plan to investigate immediately. The first is to remove memory cells that are introduced in the earliest stages of training but which are not stimulated by antigens throughout most of the training process. As indicated in Figure 3, there is a risk that such cells are not representative of the mature feature space. The second is to address the possibility that the class of the final antigen presented to the system is overrepresented in the final state of the classifier. We also expect to explore the effects of performing more than one pass through the training set. And we intend to explore the expansion of AIRS to classification problems with discrete (rather than real-valued) feature sets

IX. Conclusion

We have introduced a new and successful classifier based on the paradigm of resource limited artificial immune systems. We have described the algorithm and the principles on which the classifier is based. This algorithm is capable of performing data reduction by generating a representative set of memory cells for classification which are fewer in number than the original training instances. The same process allows the classifier to have generalized from the specific training instances. When compared against the best known classifiers for a set of four benchmark classification problems, performance is comparable to, and even exceeds performance of a number of well-known and widely used classifiers.

TABLE 1: COMPARISON OF CLASSIFIERS ON THE FOUR CLASSIFICATION TASKS. Except for AIRS data, these results are taken from [10] and [11].

	Iris		**Ionosphere**		**Diabetes**		**Sonar**	
1	Grobian (rough)	100%	3-NN + simplex	98.7%	Logdisc	77.7%	TAP MFT Bayesian	92.3%
2	SSV	98.0%	3-NN	96.7%	IncNet	77.6%	Naïve MFT Bayesian	90.4%
3	C-MLP2LN	98.0%	IB3	96.7%	DIPOL92	77.6%	SVM	90.4%
4	PVM 2 rules	98.0%	MLP + BP	96.0%	Linear Discr. Anal.	77.5%-77.2%	Best 2-layer MLP + BP, 12 hidden	90.4%
5	PVM 1 rule	97.3%	**AIRS**	**94.9%**	SMART	76.8%	MLP+BP, 12 hidden	84.7%
6	**AIRS**	**96.7%**	C4.5	94.9%	GTO DT (5xCV)	76.8%	MLP+BP, 24 hidden	84.5%
7	FuNe-I	96.7%	RIAC	94.6%	ASI	76.6%	1-NN, Manhatten	84.2%
8	NEFCLASS	96.7%	SVM	93.2%	Fischer discr. anal	76.5%	**AIRS**	**84.0%**
9	CART	96.0%	Non-linear perceptron	92.0%	MLP+BP	76.4%	MLP+BP, 6 hidden	83.5%
10	FUNN	95.7%	FSM + rotation	92.8%	LVQ	75.8%	FSM - methodology?	83.6%
11			1-NN	92.1%	LFC	75.8%	1-NN Euclidean	82.2%
12			DB-CART	91.3%	RBF	75.7%	DB-CART, 10xCV	81.8%
13			Linear perceptron	90.7%	NB	75.5-73.8%	CART, 10xCV	67.9%
14			OC1 DT	89.5%	kNN, k=22, Manh	75.5%		
15			CART	88.9%	MML	75.5%		
16			GTO DT	86.0%	SNB	75.4%		
…					. . .			
22					**AIRS**	**74.1%**		
23					C4.5	73.0%		
					11 others reported with lower scores, including Bayes, Kohonen, kNN, ID3 …			

REFERENCES:

[1] S. Forrest, S. A. Hofmeyer, and A. Somayaji, "Computer immunology," *Communications of the ACM,* vol. 40, no. 10, pp.88-96, 1997.

[2] S. A. Hofmeyr and S. Forrest, "Immunity by design: An artificial immune system," in *Proceedings of the genetic and evolutionary computation conference (GECCO),* pp. 1289-1296, 1999

[3] Jon Timmis, Mark Neal, and John Hunt, "An artificial immune system for data analysis," *Biosystems*, vol. 55, no. 1/3, pp. 143-150, 2000

[4] Jon Timmis, Mark Neal, and John Hunt, "Data analysis with artificial immune systems, cluster analysis and Kohonen networks: Some comparisons," in *Proceedings of the International Conference on Systems, Man, and Cybernetics,* pp. 922-927. 1999.

[5] Jerome Carter, "The immune system as a model for pattern recognition and classification," *J American Medical Informatics Association (JAMIA),* vol. 7, no. 1, pp. 28-41, 2000.

[6] Andrew B. Watkins, "AIRS: A resource limited artificial immune classifier," M.S. thesis, Mississippi State University, 2001.

[7] Jon Timmis, and Mark Neal, "Investigating the evolution and stability of a resource limited artificial immune system," in *Proceedings of the genetic and evolutionary computation conference (GECCO)*, pp. 40-41, 2000.

[8] Andrew Watkins and Lois Boggess, "A resource limited artificial immune classifier," (submitted).

[9] C. L. Blake and C. J. Merz, "UCI repository of machine learning databases," http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[10] W. Duch, "Datasets used for classification: Comparison of results," http://www.phys.uni.torun.pl/kmk/projects/datasets.html, 2000.

[11] W. Duch, "Logical rules extracted from data," http://www.phys.uni.torun.pl/kmk/projects/rules.html, 2000.