Oliveira, L.O.V.B.; Mota, R.L.M.; Barone, D. A C, "Clonal Selection Classifier with Data Reduction: Classification as an optimization task," *Evolutionary Computation (CEC), 2012 IEEE Congress on* , vol., no., pp.1,7, 10-15 June 2012
doi: 10.1109/CEC.2012.6256133
Abstract: This study proposes a new algorithm for supervised learning, based on the clonal selection principle exhibited in natural and artificial immune systems. The method, called Clonal Selection Classifier with Data Reduction (CSCDR), utilizes a fitness function based on the number of correct and incorrect pattern classifications made by each antibody. The algorithm tries to maximize this value through clonal selection processes such as mutation, affinity maturation and selection of the best individuals, transforming the training phase in an optimization problem. This leads to antibodies with more representativeness and thus decreases the amount of prototypes generated at the end of the algorithm. Experimental results on benchmark datasets of the UCI machine learning repository demonstrated the effectiveness of the CSCDR algorithm as a classification technique, combined with a considerable data reduction when compared to the results obtained by the well known Artificial Immune Recognition System (AIRS) and the original Clonal Selection Classifier Algorithm (CSCA).
keywords: {artificial immune systems;data reduction;evolutionary computation;learning (artificial intelligence);optimisation;pattern classification;AIRS;CSCA;CSCDR algorithm;UCI machine learning repository;affinity maturation;antibody;artificial immune recognition system;classification technique;clonal selection classifier algorithm;clonal selection principle;data reduction;fitness function;mutation;natural immune system;optimization problem;optimization task;pattern classification;supervised learning;Accuracy;Cloning;Immune system;Iris;Machine learning algorithms;Training;Vectors;Artificial immune systems;classification;clonal selection algorithm;data reduction;optimization},
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6256133&isnumber=6252855

Clonal Selection Classifier with Data Reduction: Classification as an Optimization Task

Luiz Otávio Vilas Bôas Oliveira*, Rodrigo Luiz Mendes Mota†, Dante Augusto Couto Barone*
*Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil
{lovboliveira, barone}@inf.ufrgs.br
†Institute of System Engineering and Information Technology, Federal University of Itajubá, Itajubá, Brazil
rodrigolmmota@unifei.edu.br

***Abstract*—This study proposes a new algorithm for supervised learning, based on the clonal selection principle exhibited in natural and artificial immune systems. The method, called Clonal Selection Classifier with Data Reduction (CSCDR), uses a fitness function based on the number of correct and incorrect pattern classifications made by each antibody and try to maximize this value through clonal selection processes such as mutation, affinity maturation and selection of the best individuals, transforming the training phase in an optimization problem. This leads to antibodies with more representativeness and thus decreases the amount of prototypes generated at the end of the algorithm. Experimental results on benchmark datasets of the UCI machine learning repository demonstrated the effectiveness of the CSCDR algorithm as a classification technique, combined with a considerable data reduction when compared to the results obtained by the well known Artificial Immune Recognition System (AIRS) and the original Clonal Selection Classifier Algorithm (CSCA).**

***Index Terms*—Artificial immune systems, clonal selection algorithm, classification, optimization, data reduction.**

I. Introduction

Artificial Immune Systems (AIS) is a new field of study devoted to the development of computational models based on the behavior of the biological immune system, applied to several Engineering and Computer Science problems. Some of its applications include pattern recognition, fault and anomaly detection, data analysis, agent-based systems, scheduling, machine learning, control and autonomous navigation, search and optimization methods, artificial life and information systems security [1].

Classification is a well known application of AIS. Within machine learning and pattern recognition, cases in which aim is to assign each input vector to one of a finite number of discrete categories, are called classification problems [2].

Many works have studied the use of AIS in specific classification tasks like stock market prevision [3], fraud detection [4], credit scoring [5] and clinical diagnosis [6] [7] and several other algorithms were proposed for this as the Clonal Selection Algorithm (CLONALG) [8], [9], Artificial Negative Selection Classifier (ANSC) [10], Immunos-81 [11], Artificial Immune Recognition System (AIRS) [12], [13], Weighted AIRS (WAIRS) [14], Real World Tournament Selection AIRS (RWTSAIRS) [15] and Clonal Selection Classification Algorithm (CSCA) [16].

CSCA is formulated as a function optimization procedure that maximizes the number of correctly classified patterns and minimizes the number of misclassified patterns [16]. It has been trained for several generations, and during each generation the entire set of antibodies is exposed to all antigens. The algorithm is composed by four main stages: i) Initialization, ii) Training looping, iii) Final pruning, and iv) Classification stage. In the training looping phase, antibodies are generated during initialization and they are evolved to maximize the classification rate and minimize misclassifications.

The evolving process utilizes some principles from CLONALG [8] like hypermutation and cloning. CSCA introduces a pruning process based in an user defined threshold to remove the worst antibodies. In our work, we introduced a better hypermutation process similar to that proposed in [9] and [17] and a pruning resembling the selection process from clonal selection algorithms [8], [9], [17] in order to perform a better guided search in solution space and being able to control the number of memory cells (antibodies) generated by the process. These modifications generate a new algorithm called Clonal Selection Classifier with Data Reduction (CSCDR).

CSCDR is evaluated with respect to the original CSCA and to one of best known AIS algorithms, AIRS. Both algorithms are tested with benchmark datasets from UCI machine learning repository [18] and compared with respect to the accuracy and the number of memory cells generated.

The paper is organized as follows: Section II introduces the clonal selection principle. Section III briefly describes the CSCA algorithm and its four stages. Section IV explains the modifications applied to CSCA leading to data reduction. Section V shows the proposed algorithm: Clonal Selection Classifier with Data Reduction (CSCDR) algorithm, with its description. Section VI describes the experiments and results and comparisons to AIRS and CSCA algorithms. Lastly, some conclusions are drawn in section VII.

II. Clonal Selection Principle

Lymphocytes are white blood cells produced by the bone marrow. They are specific to a wide number of antigens and exist before the exposition to them. When an antigen enters in the body, the immune system selects the specific cells and activate them. This concept is named clonal selection, and it was enunciated in [19] as a hypothesis to explain how the immune system could respond to a enormous variety of antigens.

When a antigen bind to B or T lymphocytes, they are stimulated to divide repeatedly in clone cells with the same antigen specificity from the original lymphocyte. The clones with high affinity to self antigens (harmless cells and elements from the own organism) are deleted as a provision to avoid autoimmune response.

During the process, the number of lymphocytes specific to a given antigen arises. Many of them differentiate in long life memory cells. This cells allow the immune system to respond faster and more efficiently to a second encounter with the same antigen, characterizing the immune memory.

Another concept linked to clonal selection is the affinity maturation. It is responsible for the diversification of the antibodies population in B lymphocytes. During the clonal selection, random changes are introduced into the variable region (V-region) and occasionally one such change will lead to an increase in the affinity of the antibody. These mutations occur with high rates (about one hundred thousand times greater than spontaneous mutations) and are called *somatic hypermutation* [20]. B cells with higher affinity are selected to dominate the immune response [1].

III. CSCA

For detailed information regarding CSCA, the reader is referred to literature [16]. Although its functioning principles are described here. The algorithm has four main stages described as follows:

1) *Initialization*: First there is an initial seeding phase in which randomly chosen training data vectors are used to form an initial population of antibodies.
2) *Loop*: Some steps are repeated for a determined number of generations. At each cycle, the antibodies population is presented to all training data items (called antigens) and it is verified which antigen is recognized (classified) by which antibody. The number of correct and incorrect classifications are computed only for the best match units (BMU), i.e., the antibody with biggest affinity to a given antigen. Based on this score, the fitness is calculated and used to select antibodies for cloning and mutation and to prune those with fitness lesser than a threshold value. Some remaining antibodies generated by cloning and mutation are selected to be inserted into the main antibodies population.
3) *Last pruning*: The population is exposed to the antigen population a final time, fitness scores are prepared, and pruning is performed as in step 2.
4) *Classification*: The remaining antibodies compose the set of memory cells that is used in here. During this stage data item as yet unseen by the algorithm are assigned a class based on the memory cells generated. Each new instance is compared with the centroids using a distance metric, and the closest existing instances are used to assign the class to the new one. The majority class of the closest k neighbors (defined by the user) is assigned to the new instance. This is termed the *k-nearest neighbor* (KNN) method [21], a generalization for nearest neighbor method for k neighbors.

IV. CSCA Improvement and Data Reduction

Instance-based learning methods such as nearest neighbor are conceptually straightforward approaches to approximating real-valued or discrete-valued target functions. Learning in these algorithms consists of simply storing the presented training data. When a new query instance is encountered, a set of similar related instances is retrieved from memory and used to classify the new query instance [22]. They all suffer from the same problem: the instances used to train the classifier are stored indiscriminately. No process of selection is performed, and as a result, harmful and superfluous instances are stored needlessly [23]. This may make the nearest neighbor calculation unbearably slow and may consume large amounts of storage. An apparent drawback to instance-based representations is that they do not make explicit the structures that are learned [21]. There is no explicit learning process.

However, many AIS classifiers utilize KNN method to assign a class to a new instance in classification phase. First, these algorithms learn from training data and generate a set of prototypes that represent the input data. When new instances are presented to the algorithm, a KNN method determines a classification to each new instance, based on prototypes generated. Some algorithms that implements KNN method include AIRS [12], [13] and its variations [14], [15] and CSCA [16].

During training phase, CSCA optimizes the antibodies population by selecting and evolving them to maximize the representativeness of training data. Since only the nearest neighbors are used in the classification phase, the algorithm calculates a fitness value for each antibody based solely in the already recognized data items, i.e, those items of which the antibody in question is the nearest neighbor. The fitness is calculated by (1), where $nr_{sameClass}$ is the number of recognized items from the same class of the antibody in question and $nr_{otherClass}$ is the number of recognized ones from another classes. To provide a basic population control mechanism, antibodies with fitness below a threshold value are pruned.

$$fitness(antibody) = \frac{nr_{sameClass}}{nr_{otherClass}} \tag{1}$$

Nevertheless, a static threshold based pruning may be not very effective. High values make the algorithm very selective, generating few antibodies to represent the data, which may not be able to cover the training data and low values make the algorithm memorize the data and act like a simple instance-based learning.

To overcome these issues, the threshold value is removed from the algorithm and in its place we utilize the population size parameter as a population control mechanism. As CLONALG, the population size is determined by a user defined parameter that remains limited by this upper bound. Only the antibodies with biggest fitness compose the population. Two

achievements come with this modification: the possibility of getting explicitly control the number of prototypes generated by the algorithm and the possibility of get more from less, i.e., since the population size is limited, an antibody is replaced by a new one only if its fitness is below the fitness of the new antibody, which ensures that fewer prototypes are generated at the end of the process but with greater accuracy.

V. Clonal Selection Classifier with Data Reduction

The algorithm developed here is mainly based on the CSCA, with some modifications to increase the performance and to decrease the number of memory cells generated by the process.

Clonal Selection Classifier with Data Reduction (CSCDR) generates a population of antibodies (AB) to respond to expositions to a population of antigens (AG). Antibodies and antigens are composed by a vector *d*-dimensional, representing a tuple of input data and a label, representing the class to which it belongs.

At each cycle of exposition, called *generation*, the efficiency of the response is increased by clonal expansion and somatic hypermutation. Metaphorically, antibodies are centroids that have to represent (bind) the input data (antigen) with high affinity. During training phase, these centroids are generated and evolved to be used in classification phase. In classification phase, the kNN method is applied to choose which class will be assign to each new data instance presented. The value of k is also user defined.

The training algorithm is described as:

input: Antigen set (input data)
$AB \leftarrow$ New population of antibodies
for $i = 1$ to G **do**
  Expose AB to the antigens population
  Calculate the $fitness$ of each member of AB
  Prune the worst antibodies
  $Cl \leftarrow$ Clone AB population
  $AB \leftarrow$ Mutations of Cl
end for
Prune AB for last time
return AB

- *Initialization*: A new population of antibodies is generated by choosing randomly members of antigen population. The quantity of antibodies created is controlled by an user defined parameter (S).
- *Exposure*: Each antigen is presented to the AB population. Their affinities with each member of AB are calculated and the antibody with higher affinity is selected as the *Best Match Unit* (BMU). Affinity between an antigen ag and an antibody ab is defined by opposite of the Euclidean distance calculated over their feature vectors, like is showed in (2), where d is the size of the feature space. Greater distances produce lesser affinities and vice versa. Each antibody has a counter for each possible class of the problem and when it is selected as BMU, its counter for the class of the antigen in question is incremented by one. After the exposure phase, the algorithm checks whether each antibody must change its class. If there are class counters with values greater than the count of the class itself, the antibody has its class changed to that with greater counter. This provision aims to maximize the value of fitness, calculated in the next step.

$$affinity(ab, ag) = -\sqrt{\sum_{k=1}^{d}(ab_k - ag_k)^2} \quad (2)$$

- *Fitness calculation*: (3) shows the function utilized to calculate the fitness of each antibody, similar to (1). nc is the number of instances correctly classified, defined by (4), and ni is the number of instances incorrectly classified by the antibody ab, defined by (5). $ab.counter$ is the counter utilized in last step, $ab.class$ is the index of the class to which ab belongs and N is the number of classes of the problem.

$$fitness(ab) = \frac{nc(ab)}{ni(ab)} \quad (3)$$

$$nc(ab) = ab.counter[ab.class] \quad (4)$$

$$ni(ab) = \sum_{j=1}^{N} ab.counter[j],\ j \neq ab.class \quad (5)$$

- *Pruning*: The parameter S is also used to prune AB population. In this phase, only the S antibodies with highest fitness are maintained. Other ones are removed from AB.
- *Cloning*: A population Cl of clones is generated from AB. To each member of AB, n_{clones} clones are generated, defined by (6), where β is a user defined multiplicative parameter, $round(.)$ is a function that rounds a given value to the nearest integer value, $|.|$ is the cardinality of a set and i_{ab} is the position of a antibody ab in the AB set, ordered from highest to lowest fitness value.

$$n_{clones}(ab) = round\left(\frac{|AB| * \beta}{i_{ab}}\right) \quad (6)$$

- *Mutation*: Each individual from Cl goes through a process of mutation where each component of its feature vector changes proportionally to α, defined by (7), where i is the position of the antibody in the Cl set, ordered from lowest to highest fitness and $range$ is the possible variation of a feature, i.e., the largest possible value minus the lowest possible value. A new value of an attribute att is calculated by (8), where $rnd(a, b)$ is a function that generates a random real number in interval $[a, b]$.

$$\alpha = exp\left(-\frac{3i}{|Cl|}\right) * \frac{range}{2} \quad (7)$$

$$att_{new} = rnd(att_{old} - \alpha, att_{old} + \alpha) \quad (8)$$

TABLE I
CSCDR ALGORITHM PARAMETERS

Dataset	S	G	k	β
Breast Cancer	40	20	1	1.40
Diabetes	32	25	1	1.40
Ionosphere	43	25	1	1.80
Iris	35	10	1	1.00
Glass	122	15	1	2.25
Ecoli	50	5	1	2.50

TABLE II
AIRS ALGORITHM PARAMETERS

Dataset	C	H	F	k	E	V	R
Breast Cancer	10	3	0.1	3	1	0.95	300
Diabetes	12	2	0.2	5	1	0.90	200
Ionosphere	8	2	0.3	1	2	0.95	250
Iris	9	3	0.3	5	1	0.90	150
Glass	10	3	0.1	1	2	0.90	300
Ecoli	12	3	0.2	5	2	0.95	300

- *Final pruning*: The algorithm performs one last exposure phase followed by fitness calculation. A normal pruning is realized after that followed by a pruning of the antibodies with fitness equal to zero.

One generation corresponds to one execution of exposure, fitness calculation, pruning, cloning and mutation phases, in this order. The number of generations performed by the algorithm is determined by an user defined parameter (G).

The classification task is transformed in an optimization problem in order to maximize the $fitness$ of each antibody of the population, i.e., it maximizes the number of correct classified instances and minimizes the misclassifications.

A data reduction is provided by the classifier by a user defined parameter (S) that determines the size of the antibodies population. This together with an optimization of the representation capacity of each antibody allows a generation of a defined number of centroids capable of representing the data with high rate of accuracy.

VI. EXPERIMENTS AND RESULTS

Experiments were carried in order to determine how CSCDR performed compared to CSCA and AIRS2 [13], a revised version of AIRS. Six benchmark datasets were retrieved from UCI machine learning repository [18]. Due to the inability of AIRS2 to handle datasets in which continuous and discrete attributes are present, the chosen datasets use continuous attributes only.

The algorithms parameters are optimized for each dataset. Parameters utilized for CSCDR algorithm can be found in Table I. They are described as below:

- size of AB population (S);
- number of generations (G);
- number of neighbors utilized for classification (k);
- β factor;

The following parameters of AIRS2 algorithm can be found in Table II:

- clonal rate (C);
- hypermutation rate (H);
- affinity threshold (F);
- k value in kNN classifier;
- number of seed cells (E);
- stimulation threshold (V);
- number of resources (R);

TABLE III
CSCA PARAMETERS

Dataset	k	ε	α	S	G
Breast Cancer	1	1.0	2.5	50	6
Diabetes	3	1.5	2.0	25	7
Ionosphere	1	1.0	1.5	25	8
Iris	1	1.0	2.0	25	5
Glass	1	0.5	2.0	50	9
Ecoli	1	1.0	3.0	25	6

Since the datasets tested are not very large, the number of partitions parameter utilized in CSCA is set to 1. All others following parameters of CSCA can be found in Table III:

- number of neighbors for classification phase (k);
- minimum fitness threshold (ε);
- clonal scale factor (α);
- initial population size (S);
- total of generations (G);

A 10-fold cross validation approach was utilized to estimate the predictive accuracy of the algorithms as suggested in [24]. As there is a certain amount of non-determinism both in random partitioning of the data and the running of the algorithms, the 10-fold cross validation were run 10 times for each combination of algorithm and dataset.

To determine whether a mean measure performance of an algorithm is significantly greater than, or less than, the mean of another, the *corrected resampled t-test* [25] is utilized. It is a modification of the standard *t-test* that works well in practice [21]. This modified t-test utilizes a different value of t given by (9), where n_1 is the number of instances utilized for training, n_2 for testing, k is the number of runs, σ_d^2 is the estimate of the variance of the k differences and $\bar{d}$ is defined by (10) where a_j and b_j are the values to be estimated for the algorithms A and B respectively, measured on run j ($1 \leq j \leq k$) [26]. If the means are the same, the difference $\bar{d}$ is zero (called the *null hypothesis*); if they are significantly different, the difference will be significantly different from zero. So for a given confidence level, we will check whether the actual difference exceeds the confidence limit (z).

$$t = \frac{\bar{d}}{\sqrt{\left(\frac{1}{k} + \frac{n_2}{n_1}\right)\sigma_d^2}} \tag{9}$$

TABLE IV
COMPARISON OF ACCURACY BETWEEN CSCDR, CSCA AND AIRS2

Dataset	CSCDR	CSCA	AIRS2
Breast Cancer	96.38%	96.64%	96.64%
Diabetes	72.56%	72.26%	72.26%
Ionosphere	85.96%	85.95%	85.96%
Iris	96.13%	94.07%	94.07%
Glass	69.11%	65.63%	65.63%
Ecoli	84.32%	84.02%	84.26%

TABLE V
COMPARISON OF THE NUMBER OF MEMORY CELLS LEFT AFTER TRAINING

Dataset	CSCDR	CSCA	AIRS2
Breast Cancer	40.00	76.25	307.33
Diabetes	32.00	120.92	433.33
Ionosphere	43.00	59.06	89.18
Iris	35.00	28.02	25.37
Glass	122.00	152.18	127.05
Ecoli	50.00	56.45	143.18

TABLE VI
VALUES OF t FOR COMPARISON OF MEAN ACCURACY BETWEEN CSCDR AND CSCA AND BETWEEN CSCDR AND AIRS2

Dataset	CSCDR/CSCA	CSCDR/AIRS2
Breast Cancer	-0.07605410	-0.42139000
Diabetes	0.24799448	0.14604500
Ionosphere	-0.70675780	0.00096400
Iris	0.16995438	0.91714311
Glass	-0.43898050	0.73891900
Ecoli	0.14518973	0.03717400

$$\bar{d} = \frac{1}{k}\sum_{j=1}^{k} a_j - b_j \tag{10}$$

Tables IV and V show the comparisons of mean classification accuracy and mean memory cell number, respectively, obtained when running CSCDR, CSCA and AIRS2 on the UCI datasets.

Tables VI and VII show the t values obtained by (9) for the mean accuracy and mean memory cell number. If the value of t is greater than z, or less than $-z$, we reject the null hypothesis that the means are the same and conclude that there is a significant difference between the learning methods. We adopted a confidence level of 5% which implicates in a confidence limit $z = 1.98397152$. Significant results are written in bold type.

From Tables IV and VI, it can be seen that there is no significant difference in performance between CSCDR and others algorithms. CSCDR is not a special algorithm for a particular problem, but a general purpose algorithm as AIRS and CSCA. Thus, accuracy rates similar to those obtained by these algorithms are expected.

Nevertheless, from Tables V and VII, it can be seen that, except for the Iris dataset, the mean memory cell number generated by CSCDR algorithm are significantly less than the other algorithms for the tested sets. This is due to the high representativeness of the prototypes generated by the algorithm that on average are able to describe more data of the problem space.

TABLE VII
VALUES OF t FOR COMPARISON OF MEAN MEMORY CELL NUMBER BETWEEN CSCDR AND CSCA AND BETWEEN CSCDR AND AIRS2

Dataset	CSCDR/CSCA	CSCDR/AIRS2
Breast Cancer	**-13.60556821**	**-136.00650330**
Diabetes	**-39.25942938**	**-121.49044510**
Ionosphere	**-12.04322254**	**-24.82455548**
Iris	**5.53497968**	**12.01279473**
Glass	**-15.36249221**	**-2.88492674**
Ecoli	**-4.75077779**	**-43.32701349**

VII. CONCLUSIONS

In this work, we modified CSCA improving the mutation process by utilizing an exponential function and inserting a control parameter for the number of memory cells produced. The former helps in search process and the later forces the algorithm to generate antibodies with greater space coverage, providing a good trade off between their classification rates and their misclassifications.

CSCDR shows good accuracy for benchmark datasets as for AIRS and CSCA and high data reduction of prototypes utilized for classification phase. This reduction decreases the memory cells storage and decreases processing time during the classification step. Besides that, as consequence of the modification of CSCA, developed CSCDR has only four input parameters, against five from the original algorithm and seven from AIRS. A smaller number of parameters reduces the search space when looking for an optimal combination of parameters.

The pruning method maintain antibodies with greatest fitness values, reducing the effect of noise and reducing the danger of overfitting. Future works might investigate CSCDR algorithm with noisy datasets.

Another possibility of future work is the investigation of an automatic process of pruning, without a control parameter.

REFERENCES

[1] L. N. de Castro and J. Timmis, *Artificial immune systems: a new computational intelligence approach*. Springer-Verlag, 2002.

[2] C. Bishop, *Pattern recognition and machine learning*, ser. Information science and statistics. Springer, 2006.

[3] M. Butler and D. Kazakov, "Modeling the behavior of the stock market with an artificial immune system," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, july 2010, pp. 1–8.

[4] A. Brabazon, J. Cahill, P. Keenan, and D. Walsh, "Identifying online credit card fraud using artificial immune systems," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE, 2010, pp. 1–7.

[5] K. Leung Kan Hing, F. Cheong, and C. Cheong, "Consumer credit scoring using an artificial immune system algorithm," in *CEC 2007*. IEEE, 2011, pp. 3377–3384.

[6] W. Zhao and C. Davis, "A modified artificial immune system based pattern recognition approach–an application to clinical diagnostics," *Artificial Intelligence in Medicine*, 2011.

[7] K. Delibasis, P. Asvestas, G. Matsopoulos, E. Zoulias, and S. Tseleni-Balafouta, "Computer-aided diagnosis of thyroid malignancy using an artificial immune system classification algorithm," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 5, pp. 680–686, 2009.

[8] L. N. de Castro and F. J. Von Zuben, "The clonal selection algorithm with engineering applications," in *GECCO'00 - Workshop Proceedings*, 2000, pp. 36–37.

[9] L. N. de Castro and F. Von Zuben, "Learning and optimization using the clonal selection principle," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 3, pp. 239–251, 2002.

[10] K. Igawa and H. Ohashi, "A negative selection algorithm for classification and reduction of the noise effect," *Applied Soft Computing*, vol. 9, no. 1, pp. 431–438, 2009.

[11] J. Carter, "The immune system as a model for pattern recognition and classification," *Journal of the American Medical Informatics Association*, vol. 7, no. 1, pp. 28–41, 2000.

[12] A. Watkins, "AIRS: A resource limited artificial immune classifier," Master's thesis, Mississippi State University, 2001.

[13] A. Watkins, J. Timmis, and L. Boggess, "Artificial immune recognition system (AIRS): An immune-inspired supervised learning algorithm," *Genetic Programming and Evolvable Machines*, vol. 5, no. 3, pp. 291–317, 2004.

[14] A. Secker and A. Freitas, "WAIRS: Improving classification accuracy by weighting attributes in the AIRS classifier," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. IEEE, 2007, pp. 3759–3765.

[15] S. Golzari, S. Doraisamy, M. Sulaiman, and N. Udzir, "Improving the accuracy of airs by incorporating real world tournament selection in resource competition phase," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 3040–3044.

[16] J. Brownlee, "Clonal selection theory & CLONALG - the clonal selection classification algorithm (CSCA)," Centre for Intelligent Systems and Complex Processes (CISCP), Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology, Tech. Rep. 2-02, 2005.

[17] L. N. de Castro and J. Timmis, "Artificial immune systems as a novel soft computing paradigm," *Soft Computing*, vol. 7, no. 8, pp. 526–544, 2003.

[18] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[19] F. Burnet, "A modification of jerne's theory of antibody production using the concept of clonal selection," *Australian Journal of Science*, vol. 20, no. 3, pp. 67–69, 1957.

[20] A. Abbas, A. Lichtman, and S. Pillai, *Cellular and molecular immunology*, 6th ed. Saunders Elsevier, 2007.

[21] I. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. Morgan Kaufmann, 2005.

[22] T. Mitchell, *Machine Learning (Mcgraw-Hill International Edit)*, 1st ed. McGraw-Hill Education (ISE Editions), 1997.

[23] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," *Data mining and knowledge discovery*, vol. 6, no. 2, pp. 153–172, 2002.

[24] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International joint Conference on artificial intelligence*, vol. 14. Citeseer, 1995, pp. 1137–1145.

[25] C. Nadeau and Y. Bengio, "Inference for the generalization error," *Machine Learning*, vol. 52, no. 3, pp. 239–281, 2003.

[26] R. Bouckaert and E. Frank, "Evaluating the replicability of significance tests for comparing learning algorithms," *Advances in knowledge discovery and data mining*, pp. 3–12, 2004.