IMMUNOS-81

THE MISUNDERSTOOD ARTIFICIAL IMMUNE SYSTEM

Technical Report
No. 3-01

Jason Brownlee
Master of Information Technology, Swinburne University of Technology, 2004
Bachelor of Applied Science, Computing, Swinburne University of Technology, 2002
(jbrownlee@ict.swin.edu.au)

Centre for Intelligent Systems and Complex Processes (CISCP)
Faculty of Information & Communication Technologies (ICT)
Swinburne University of Technology (SUT)

January 2005

Abstract

The vertebrate immune system is a robust and powerful information process system that demonstrates features such as distributed control, parallel processing and adaptation and learning via experience. Artificial Immune Systems (AIS) are machine-learning algorithms that embody some of the principles and attempt to take advantages of the benefits of natural immune systems for use in tackling complex problem domains. The Immunos-81 is an AIS technique designed for classification problem domains. It is a technique that has been vaguely described and mentioned in the field of AIS research, though has not been investigated in depth nor has the algorithm or its results been reproduced. This work rigorously analyses the proposed classification system and describes both its biological inspiration and its computational implementation in detail. Two implementations are provided, that reproduces the general themes of the approach and show similar results. Finally, the general themes of the system are integrated with elements from clonal selection inspired algorithms. A new classification algorithm is designed, implemented and tested called Immunos-99 that exhibits desirable characteristics such as excellent data reduction and moderate classification accuracy, and remains an area for further research.

Acknowledgements

Firstly, I would that to sincerely thank my advisor Professor Tim Hendtlass and the CISCP for giving me the opportunity and financial support to research and work in a field that is both interesting and practical, and something I am passionate about. Secondly, thanks to the support provided by all the people at CISCP, for listening to my rants and for consistently providing positive and constructive feedback. Thanks to Andrew Watkins and Dr. Simon Garrett for discussing various elements of the Immunos-81 system. Finally thanks to Dr. Warren Jones from the University of Alabama for assisting me in attempting to get into contact with Dr. Jerome Carter.

Table of Contents

List of Figures

List of Equations

List of Tables

1. Introduction

Artificial immune systems are a technique new to the scene of biological inspired computation and artificial intelligence, based on metaphor and abstraction from theoretical and empirical knowledge of the vertebrate immune system. A robust biological process critical to the combating of disease in the body, the immune system is known to be distributed in terms of control, parallel in terms of operation, and adaptive in terms of function, all of which are features desirable for solving complex or intractable problems faced in the field of artificial intelligence.

Immunos-81 is an immune system inspired algorithm for pattern recognition and classification. It was devised by Carter [1], (a medical doctor). It is an artificial immune system that is typically misunderstood and incorrectly described in AIS literature, given the medical-based as opposed to computer science based description of its implementation. The goal of this work is to provide a detailed analysis of the work in which Immunos-81 is described and provide a description and implementation of the technique that is capable of reproducing the successful results reported of the technique. Further, the intent of this work is postulate as to the success of the technique, test the approach on additional problem domains and proposed extensions to technique which may prove useful in practical application. A new Immunos-inspired classification algorithm called Immunos-99 is proposed and tested. It is shown to be an amalgamation of the desired elements from the Immunos system and those from other clonal selection-based classification algorithms.

The immunos-81 algorithm has been mentioned a number of times in AIS literature [2-5]. It is claimed as being one of the first immune-inspired classification systems, though to the author's knowledge, the work has not been investigated in depth nor had its results reproduced to date. Commonly the descriptions of the approach are vague or inaccurate given likely misinterpretations of the original work. A possible explanation of the lack of reproduction and accurate dissection of the work is due to both the complexity of the proposed system and the algorithm descriptions provided in a manner not typical for computer science publications.

"… it was devised by a medical professional, was highly complex, and was not sufficiently described to replicate." [5]

The goal of this work is to sufficiently describe the algorithm based on the single work on the topic, and to attempt to reproduce the result based on the new understanding. It should be noted that many attempts were made to track down Dr. Jerome Carter to discuss his Immunos-81 technique and clarify points; though at the time of competing this report Carter was not successfully reached.

Section 2. covers the biological background for the Immunos-81 system as described by Carter. Those elements that are abstracted and subsumed by the algorithm are described in detail, as are the design goals for the system. Section 3. provides a review of the terminology used to describe the system and provides a detailed review of the algorithms

procedure based on a rigorous review of Carter's work. Section 4. proposes two implementations of the generalised Immunos-81 system called Immunos-1 and Immunos-2. Results for the interpreted Immunos systems indicate similar results to those reported by Carter. Further testing on additional domains reveals the algorithm is capable of performing well in some instances, though required additional work for practical application. Section 5. examines the potentially beneficial elements of Immunos and proposes a new algorithm called Immunos-99 which both extends the general Immunos technique and integrates elements from other AIS algorithm inspired by the clonal selection theory of acquired immunity. Results indicate excellent data reduction capabilities with similar though improved results compared to the Immunos-1 and Immunos-2 systems. Considered an algorithm under development, the Immunos-99 system is a technique that requires further investigation harness its power for practical application.

2. Biological Background

Carter provides a detailed account of the elements of the natural immune system that inspire the work on Immunos-81, likely given the training of a medical doctor. This section provides an overview of the interesting and relevant portions of that review. Further, this section provides a review of the design goals for Immunos-81 specified by Carter.

2.1 Inspiration for Immunos-81

The mammalian immune system is an organ whose role it is to protect the host organism from potentially harmful materials (pathogens). This is achieved by specialist cells recognising those molecules that are foreign and neutralising them. An important point is that through experience with antigens (molecules that evoke an immune response); the system is able to improve itself, adapting to provide an increasingly stronger and more rapid response. There are two main types of recognition cells involved in this process called T lymphocyte cells and B lymphocyte cells (T-cells and B-cells).

T-cells are called such because they develop in the thymus (a glandular structure that functions in the development of the immune system). The development of T-cells involves a two-step selection process:

1. Positive Selection – involves the entire population of T-cells, and only those cells that functionally demonstrate their recognition capability are maintained, the reset undergo programmed cell death (apoptosis)
2. Negative Selection – the cells the survive the first step are exposed to self-antigens. Those cells that recognise the antigens also undergo cell death

Approximately 98% of T-cells do not make it through the selection process, though it is estimated that 10^6 T-cells make into circulation each day. B-cells mature in the bone marrow and are produced at a similar rate as T-cells. The distributions of each type of cells throughout the body differ. Approximately 90% of T-cells circulate; where as 90% of B-cells are found in secondary lymph tissues (such as the lymph nodes, spleen and tonsils).

When a T-cell or a B-cell encounters an antigen, and has a sufficient affinity with its surface receptors, the cell becomes activated. The cell binds to the antigen though this step alone is not sufficient to elicit an immune response. B and T-cells that recognise the same antigen must have direct physical contact with each other which then permits both cells to proliferate (divide) into a clone (group of cells) of cells with identical receptors as the parent cell. The size of the clone seems proportional to the degree of match or affinity between the antigen and the cells receptors, meaning the cell with the highest affinity produces the most progeny. B-cells undergo an additional transformation called affinity maturation. This is an element of the system that allows some of the clone of B-cells to improve their affinity with the antigen via point mutations at sites responsible for antigen binding. From the cell division process, memory cells are produced. These are T-cell or B-cells that remain in the organism for months or years and are able to elicit an increased response for the antigen for which they are specialised.

Those antigens that evoke the largest response are composed of proteins, which are large molecules that are made from smaller molecules called amino acids. Although amino acids exhibit three types of organisation, only the primary and tertiary forms are of interest to this discussion. The primary structure of the protein refers to the physical sequence of amino acids that is the order, whereas the tertiary structure is the three-dimensional structure of the protein with all of folds and twists. The relevance of this information about proteins is that T-cells learn the antigens primary structure, and B-cells learn its tertiary structure. This is clearly an excellent example of the systems ability to both delegate responsibility and process information in parallel.

A final note concerns the Idiotypic Network Theory (INT) proposed by Jerne. This is a theory for describing the dynamics of the interactions between elements of the immune system and proposes that the immune system itself is regulated by antibody-antibody and antibody-lymphocyte interactions. Specifically the theory describes the systems ability to achieve and maintain a state of equilibrium when not responding to an antigen. This requires the system to generate a regulating force or counter idiotypic response when lunching a defensive against an antigen. Although other artificial immune systems use the theory as the basis for algorithm design, Immunos-81 does not model idiotypic interactions.

2.2 Design goals

Carter makes the argument that the majority of AIS reviewed at the time adhere too closely to the biological metaphor, which although provides some useful architectural algorithm elements may not necessary be useful from a computational point of view. The example of the back-propagation artificial neural network algorithm is provided where although it does not model a process of biological neurons it uses elements of the metaphor and is widely used and successful. In the biological immune system, millions of cells are produced each day, and most cells die on contact with the antigen, both elements of which are expected to be less than useful in an artificial immune system. The primary design goal for Immunos-81 is to reduce elements of the immune system to the basic concepts for practical application.

Carter proposed nine concise design goals for Immunos-81, which were not elaborated on, as follows:

1. Easily understood internal representation
2. Ability to generalise from input data
3. Predictable training times
4. Online learning
5. Potential to act as associative memory
6. Acceptance of continuos and nominal variables
7. Capacity to learn and recall large numbers of patterns
8. Experience-based learning
9. Supervised learning

The primary reasons why the immune system was selected as the basis for a classification algorithm was given its ability to readily accept patterns of arbitrary length, and its ability to maintain and exploit previously learned information efficiently for improved performance in future encounters.

3. The Immunos-81 Classification System

As mentioned, the description of the technique is somewhat incomplete or lacking detail. This section provides terminology used to describe the system in the context of immunology. Further, a description of the Immunos-81 system is provided based on a rigorous review of Carter's work [1].

3.1 Algorithm Terminology

Carter uses a immune-inspired terminology is used to describe the algorithm, and thus the same terminology is used here. Some modifications to this terminology have been made to ensure that the algorithm description is concise, though the components or processes they relate to in the algorithm remain the same.

T-cell – These elements learn the primary structure of antigens in the system and decide what is interesting to the system. The T-cells act as gatekeepers to the system, both partitioning learned information and deciding how new information is exposed to the system. One T-cell exists for each antigen-type in the system, and each T-cell has one or more clone (groups) of B-cells.

B-cell – These elements learn the surface features of antigens. In the system, B-cells represent an instance of an antigen-type, more specifically an instance of an antigen-group. It is mentioned by Carter that B-cells are not represented explicitly by Immunos-81, rather they are incorporated into a clone or group structure.

Antigen – An antigen is a molecule that elicits an immune response, which in this case is represented by a single training data instance. An antigen is represented as a data vector of attribute or field values where the nature of each attribute (name, data type) are known.

Antigens can be variable length, and can have missing field values. Further, Immunos-81 supports the acceptance of antigens from differing problem domains.

Antigen-type – Confusingly called antigen-class by Carter, the antigen-type refers to a specific problem domain identified by a specific name and series of attributes. This is the primary structure learned by T-cells.

Antigen-group (clone) – An antigen group is a set of antigens of a specific antigen-type that are of interest. Given that the nature of the problem addressed by Immunos-81 is that of classification, an antigen-group represents all antigens with a specific classification label. Represented in the system, this group is called a clone of B-cells controlled by a single T-cell.

Concentration (antigenicity) – An interesting term used to describe the number of antigens (training instances) with a specific class label (an antigen-group) was antigen concentration or antigenicity. This concentration obviously indicates the amount of attention each antigen-group will receive, as well as aids in defining the size of the antigen-groups clone of B-cells.

Paratope and Epitope – A paratope is a part of the receptor on an antibody molecule that binds to a matching part of the antigen molecule called the epitope. In Immunos-81 these terms are used to described individual attributes on the recognition cells (T-cells and B-cells), and antigens respectively.

Affinity – A recognition cell has specificity with an antigen, the degree that is called affinity. In the Immunos-81 system, affinity is a match between a similarity/dissimilarity measure between a T-cell and an antigen, and a B-cell and an antigen.

Avidity – Avidity is the term used to describe the sum affinities for all B-cells of a clone for a given antigen. This value is used to allow clones of a T-cell to compete with each other to classify an antigen.

Amino Acid Library (AALib) – As mentioned, antigens that exhibit the largest immune response consists of proteins, which are sequences of amino acids. Immunos-81 uses a similar concept to manage antigen details. A given-antigen type has a name (problem domain name) and a series of attributes (order, attribute names, data types, etc.). This information is required for a T-cell and throughout the learning process. All this domain knowledge is centralised and stored in an AALib construct.

3.2 Algorithm Description

Although Carter describes his system at great length regarding the biological inspirations and abstractions, some specifics of the algorithm were omitted from the single publication. Given this concern, this section proposes a generalised Immunos-81 implementation based on a rigours review of the available work and may be open to specific interpretation.

3.2.1 Initialisation

Little if any initialisation is required for applying Immnous-81 in practice. The system is does not require seeding with an initial T-cell population or B-cell clones. One useful feature of the system raised though not explored by Carter is the ability for the system to be used in a dynamic or online-learning environment. This implies that it may be useful to initialise the system with a previously prepared Immunos-81 classifier, taking advantage of all elements of the system including the AALib, the T-cell population and each T-cells prepared clones.

3.2.2 Training

The training specification of the system assumes that all training data instances are available at training time. Further it requires that instances are grouped first by antigen-type and then by antigen-group and exposed to the system one antigen-group at a time.

The following specifies the training schedule:

1. Group antigens into antigen-types
2. For each antigen-type
    a. Prepare a T-cell to represent the antigen-type
    b. Group into antigen-group
    c. For each antigen-group
        i. Prepare a Clone of for the group allocated to the current T-cell

The creation of the B-cell population (clone) is a critical aspect of the algorithm as the clone is the basic recognition unit of the system. The clone is prepared in two steps:

1. Prepare Clone Population
    a. A single B-cell is created for each antigen in the antigen-group
    b. The created B-cells are capable of responding specifically to the antigen that caused its creation
    c. The size of the population defines the antigenicity of the clone
2. Prepare Clone Proper
    a. After exposure to all antigens in the antigen-population the system is considered to have learned all the features that describe the group
    b. A condensed representation of the clone population is prepared
    c. During this process, details of each paratope are recorded such as min, max and mean for numeric attributes, and frequencies for nominal attributes.
    d. The specifics of the consolidation are not provided, though it is clear that the clone population is abandoned after the condensed representation is prepared

The training scheme can be generalised to the following schematic representation:

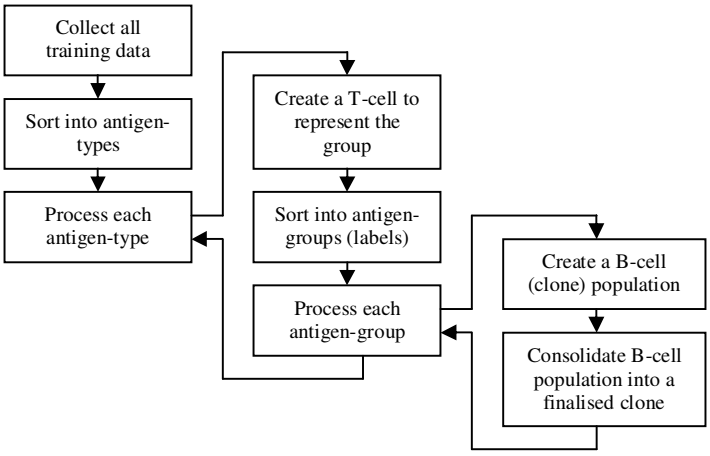

Figure 1 - Generalised version of the Immunos-81 training scheme

During the finalising of the clone (consolidation of B-cells), affinity and avidity scores are calculated for the clone. As mentioned, the specifics of the process are not defined, though the following provides an overview of the process:

1. Affinity Calculation - Involves the calculation of affinity values for each individual paratope (attribute) across all B-cells in the clone-population. These individual paratope affinity values are scaled, the direction of which is not specified. The following equation defines paratope affinity scores:

$$pa_i = k \cdot \sum_{S}^{j=1} a_j$$

Equation 1 – Training paratope affinity calculation

where pa_i is the paratope affinity for the i^{th} paratope, k is the scale factor, S is the total B-cells in the population, and a_j is the affinity for the j^{th} B-cell in the clone population. The specifics of what the a term represents were not provided. There is an indication from Carter's work that the condensed representation may involve affinity values in some way.

2. Avidity Calculation – Avidity values are calculated for each clone during the training schedule, though the purpose of the derived value is not indicated. The avidity score is calculated as the sum of the affinity values scaled both by the size of the clone population and an additional scale parameter, as follows:

$$ca_i = k_{2i} \cdot \left(\sum_{N}^{j=1} pa_j \right) \cdot S_i$$

Equation 2 - Training avidity calculation

where ca is the clone avidity for the i^{th} clone, k_{2i} is a user parameter used to scale avidity (specific to the i^{th} clone) based on the number of clone for the antigen-type, pa is the paratopic affinity for the j^{th} paratope, N is the total paratopes, and S_i is the total number of B-cells in the i^{th} clone population.

3.2.3 Classification

Classification in Immunos-81 is a competitive two-step process as follows:

1. Match T-cell – The antigen is compared to all known T-cells in search of an exact match. Once located, the clones for the T-cell are retrieved for the second step of the classification process.

2. Select Best Clone – The antigen is then exposed to each clone of the matching T-cell. Affinity values are calculated for each antigen epitope and clone paratope combination. The affinity scorings are summed to provide an avidity value that represents the clone's response to the antigen. It is unclear if the affinity and avidity scores calculated within each clone the same as those used during training. The unknown antigen is then assigned a class label in one of two ways:

1. Simple Highest Affinity (SHA) – the clone with the highest affinity is selected
2. Relative Highest Affinity (RHA) – the clone with the highest affinity is selected, if the delta between the best clone and second best clones avidity is at least 5% greater then the classification label is assigned, otherwise the label of "too close to determine" is assigned

The clone avidity ca values calculated for this stage are continuous and in the range $ca \in [0,1]$. The classification scheme can be summarised using the following figure:

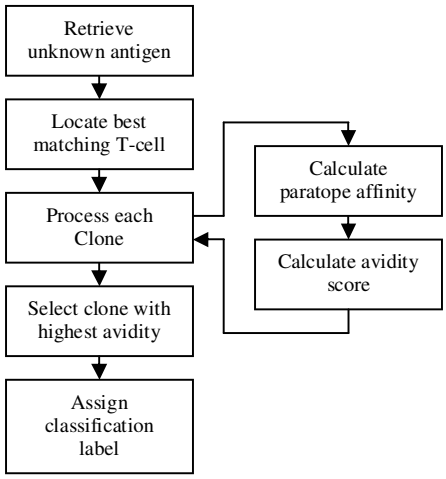

Figure 2 - Overview of the Immunos-81 classification scheme

4. Interpreted Immunos-81

Given that insufficient information is provided to completely reproduce an implementation of the Immunos-81 system, an implementation of the system can be interpreted or speculated at. These proposed implementations of the technique are based on assumptions as to the source of power of the technique and elements of the technique, which, clearly differentiate it from other more common artificial immune systems.

4.1 Features

This section speculates as to the beneficial and useful feature of Immunos-81 that may be the source of the techniques success and may be generally useful or desirable for AIS algorithms.

Generalisation – The system provides generalisation capability through data reduction. The antigen-group that is used to prepare a B-cell clone population is eventually reduced into a condensed representation of the features of the entire B-cell population.

Variable length vectors – The system supports variable length antigen vectors, more meaningfully, this means that the system supports learning for multiple problem domains in parallel.

No seed population – Unlike other AIS techniques, Immunos-81 does not require a seed population. Instead, the system prepares specific elements in response to the data it is exposed to, as it is exposed to it.

Single training iteration – The system does not require multiple exposures to the training data as is seen in other AIS systems. This efficiency permits the system to prepare itself via a single pass or observation of the training data, a feature desirable for rapid learning, particularly in an online or dynamic environment.

Dynamic / online learning – The system has the capacity to be used in a dynamic fashion for online learning. Although the current training scheme does not lend itself to such, it is likely that with a reordering of the scheme (such as accumulating B-cell populations on the fly and running peridotic consolidation phases), that the system can be applied and tested in dynamic environments.

Confidence level – In applying a threshold difference between clone avidity values when competing to classify an unknown instance permits the user to tune the system to a specific pre-established confidence level. Alternatively, this difference can be provided with classification decisions to indicate the systems confidence in its decision. Further, the ability to indicate an "I don't know" classification is a useful feature of the system, in that it can be used as an indicator of the areas of the problem domain that the system needs further attention.

Competition – Ultimately, the algorithm is a competitive environment, which is a powerful tool for both accelerating the effectiveness and providing robustness to a

system. It permits all elements to participate in the decision making process, though rewards the element better than all others by assigning a classification label.

When classifying a single data instance, Immunos-81 makes the decision based on two main precepts:

1. maximise receptor affinity between the antigen and the clone
2. maximise concentration between the antigen and the clone

These rules are the core to the intent of the techniques definition and are expected to be realised in any Immunos-based implementation.

4.2 Naïve Immunos Algorithms

Given the description of Immunos-81's training procedure and an identification of the potentially useful and desirable elements of the system, it is possible to propose an implementation of the technique. It is not expected that the implementation match exactly the partial information provided by Carter, rather the focus is on repeating the intent and primary function of the system. This section describes two basic implementation of the Immunos system, with the goal of repeating the results observed in the original work.

4.2.1 Immunos-1

The first implementation of Immunos assumes no data reduction, thus the clone population prepared is maintained and is used to classify unknown data instances. This naïve approach is provided as a baseline for performance, and is very similar to the k-nearest neighbour algorithm. The primary difference is obviously that the training population is partitioned, and *k* is set to one for each partition.

Training of the system assumes a single problem domain. This simple change eliminates T-cells and the need for an Amino Acid Library (AALib). Multiple-problem support can be provided with a simpler mechanism that uses a classifier for each problem, and a dispatcher to funnel training instances to the specific problem as follows:

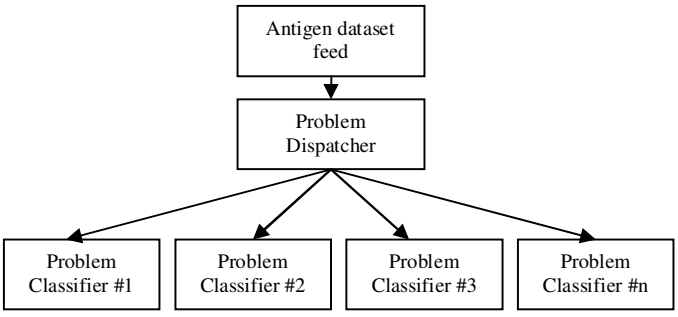

Figure 3 - Dispatcher based configuration for supporting multiple problem domains in parallel

This obvious redefinition for supporting multiple problem domains is more flexible in that it permits both independent parallel management of each classifier, dynamic addition

and subtraction of classifiers (domains) and classifiers of varying types. These are only a few of the beneficial features of such an implementation. The configuration is also reminiscent of previously proposed parallel AIS systems such as Parallel-CLONALG [6] and Parallel-AIRS [7], only the proposed structure is parallel for problem domains instead of a single domain. The area of multiple-problem support provided by Immunos-81 will not be investigated further in this work, and as mentioned, all Immunos-inspired implementations assume a single problem domain, and thus a single classifier.

A B-cell population is created for each feature of interest in the problem domain; in the case of classification, this means one population per known classification label. A single B-cell is created for each antigen in the training set, and is added to the population representative of the instances class label.

During classification, the prepared populations of B-cells compete to label unknown data instances. This is achieved by calculating an avidity value for each clone population, then assigning a classification value of the clone population with the highest avidity. Avidity is calculated in a two-step process. Firstly, the affinity between the antigen in question is taken for all members of the clone population. Euclidean distance is used as an affinity measure between data vectors. In the case of nominal attributes, a binary distance value is calculated (zero for match, one for no match).

The following equation is used to calculate affinity between a single B-cell and an unknown antigen in question:

$$affinity = \sqrt{\sum_{A}^{i=1} af_i}$$

Equation 3 - Affinity calculation between a B-cell and an antigen

where A is the total number of attributes in the data vectors and af_i is the affinity of the i^{th} attribute. As mentioned, affinity for nominal and numeric attributes is handled differently, as follows:

$$numeric = (ab_i - ag_i)^2$$

Equation 4 - Affinity calculation for a single numeric attribute

where ab_i is the i^{th} attribute on the B-cell, and ag_i is the i^{th} attribute on the antigen.

$$nominal = \begin{cases} 0 & if \quad (ab_i \equiv ag_i) \\ 1 & if \quad (ab_i \neq ag_i) \end{cases}$$

Equation 5 - Affinity calculation for a single nominal attribute

The second and final step for calculating avidity is to incorporate the size of the clone into the equation. This conforms to a founding principle of Immunos-81, namely the affinity and the antigenicity of a population define its response (are used to declare the winning clone population). The following equation represents the final avidity calculation:

$$avidity = \frac{CS}{CA}$$

Equation 6 - Final avidity calculation for a single clone population

where CS is the Clone Size, that is the total number of B-cells in the clone in question, and CA is the Clone Affinity, that is the summed affinity for the clone and the antigen calculated by applying Equation 3 to all B-cells in the clone population.

Given that the affinity calculation provided in a distance value (meaning smaller values have higher affinity), Equation 6 matches the behaviour exhibited by Immunos-81 in Equation 2. That is the system rewards high affinity values and high concatenation. Note that a clone-specific scale value could be used in this equation, though was intentionally omitted for simplicity reasons.

4.2.2 Immunos-2

The Immunos-2 implementation is the same as Immunos-1, only it seeks to provide some form of basic generalisation via data reduction, and thus a closer representation to the original Immunos-81 proposal. Clone populations are constructed as in Immunos-1, though an additional set is performed that reduces each B-cell population to a single exemplar for use during classification. This permits the prepared B-cell populations to be discarded after training as in Immunos-81.

The single B-cell exemplar for each clone population is prepared by simply (and naïvely) taking the mean attribute value across the entire population. In the case of nominal attributes, the nominal value with the highest frequency is taken for the exemplar.

Classification occurs in a similar manner as in Immunos-1, though affinity for a presented unknown antigen is calculated using only the prepared exemplar. The avidity calculation from Equation 6 is still used, as are the same affinity equations used in Immunos-1.

4.3 Reproduction of Results

In providing an implementation of the Immunos-81 technique, the intent is to reproduce previous results reported by Carter. Carter performed only two tests of the system, both involving the Cleveland Heart Disease (CHD) dataset [8]. The tests were performed using the SHA and RHA techniques for selecting a clone winner, though due to the implementation details of Immunos-1 and Immunos-2, only SHA results are considered. Note that implementing RHA for the two algorithms for testing could be achieved easily. This was not done given the focus of this section was to demonstrate an accurate understanding of the basic technique.

The first test involved performing a 10-fold cross-validation test on the CHD dataset, which a result of 83.2% accuracy was achieved for SHA. The second test used the CHD dataset as training data, and used the Long Beach Veterans Administration (VA) dataset for testing.

Only the first test performed by Carter was be repeated. In addition, the naïve Immunos implementations were tested on a number of other standard machine learning datasets. This was provided as a rough guide as to the effectiveness of the underlying Immunos principles for a classification system. Datasets were selected for a mixture of nominal and numeric attributes and varied number of attributes. Each cross-validation test was repeated 10 times, and provided with mean, standard deviation values. It should be noted that all datasets were normalised to unit length before presentation to the algorithm. The following lists the datasets used for testing and the number of folds in used for cross-validation. Cross-validation values come from [9,10] and are used for comparison reasons.

Selected Datasets

Dataset	Cross-validation folds
Cleveland Heart Disease (CHD)	10
Iris Plants (IP)	3
Sonar (S)	13
Wisconsin Breast Cancer (WBC)	10

Table 1 - Datasets used to test Immunos algorithm implementations

The following provides the test results on the selected datasets:

Immunos-1

Dataset	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Mean	Stdev.	Min	Max
CHD	81.518%	80.528%	81.188%	80.858%	81.188%	80.858%	81.188%	80.858%	80.858%	80.858%	**80.99%**	0.264	80.528%	81.518%
IP	95.333%	97.333%	96.667%	97.333%	97.333%	97.333%	97.333%	97.333%	98%	97.333%	**97.133%**	0.67	95.333%	98%
S	68.269%	67.788%	66.827%	67.788%	68.75%	66.346%	68.269%	68.75%	69.231%	67.308%	**67.933%**	0.861	66.346%	69.231%
WBC	86.123%	86.409%	86.409%	85.837%	85.98%	86.123%	86.123%	86.266%	85.837%	86.266%	**86.137%**	0.197	85.837%	86.409%

Table 2 - Immnos-1 results for selected datasets

Immunos-2

Dataset	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Mean	Stdev.	Min	Max
CHD	80.858%	80.198%	80.528%	80.528%	80.528%	80.528%	80.528%	80.528%	80.528%	80.528%	**80.528%**	0.148	80.198%	80.858%
IP	94.667%	97.333%	96.667%	97.333%	97.333%	97.333%	97.333%	97.333%	97.333%	97.333%	**97%**	0.803	94.667%	97.333%
S	61.058%	62.5%	61.058%	62.019%	61.058%	61.058%	62.019%	62.5%	61.538%	61.538%	**61.635%**	0.561	61.058%	62.5%
WBC	68.813%	68.67%	68.67%	68.67%	68.813%	68.67%	68.67%	68.67%	68.813%	68.956%	**68.741%**	0.096	68.67%	68.956%

Table 3 - Immnos-2 results for selected datasets

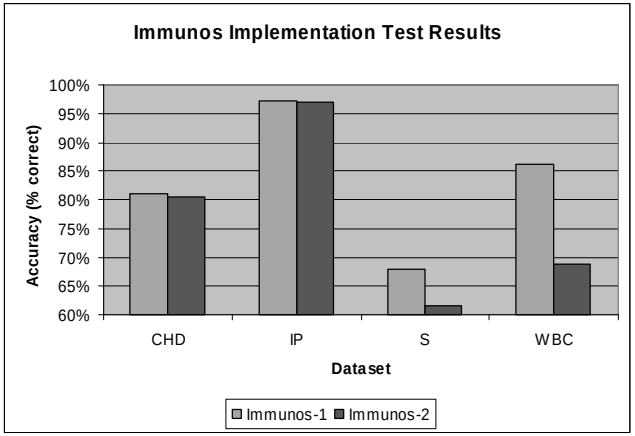

Figure 4 - Shows the mean test results for the two immunos implementations

4.4 Discussion

The results for both Immunos implementations on the Cleveland dataset were shown to be close to those reported by Carter for Immunos-81, showing a difference of approximately 2.2% accuracy for the better of the two implementations; Immunos-1. It should be noted that tests were repeated 10 times to provide a statistically more meaningful figure. It should also be noted that the results for the proposed implementations are still within the top 20 classifiers on Duch [9] for the CHD dataset. Given the closeness of the results, it can be speculated that the core elements of the Immunos-81 system are present in the proposed Immunos-1 and Immunos-2 implementations.

Comparing the two algorithms from the results provided in Table 2, Table 3 and Figure 4, it is clear that the Immunos-1 implementation provided improved results. This is highlighted particularly with the S and WBC datasets that showed a 6.3% and 17.4% difference for mean classification accuracies respectively. Results on the Iris Plants dataset were impressive indeed. When compared to classifiers on Duch [10], both implementations are placed second.

The two implementations of Immunos were designed as proof-of-concept algorithms to demonstrate features of Immunos-81 and attempt to reproduce previously observed results. When comparing the implementations to other non-immune inspired classification algorithms, they resemble (though not precisely) techniques such as k-Nearest Neighbour and other instance based classifiers. Given the success in both close-to repeating original results, and in classification accuracy on other datasets, particularly the Iris Plants the principles of the Immunos-81 algorithm deserve further attention.

5. Immunos-inspired Algorithm (Immunos-99)

Given the success in interpreting the Immunos-81 specification and implementing two variants of the technique, a new Immunos-inspired algorithm is designed called Immunos-99. The proposed algorithm exploits those elements from Immunos-81 that appear beneficial and unique to the system, and integrates cell-proliferation and hypermutation techniques from other immune-inspired classification systems. The algorithm is specified in detail and tested against the same datasets as Immunos-1 and Immunos-2. Results indicate superior data-reduction capabilities and a general insensitivity to algorithm parameters. The new hybrid immune classification system shows promise in practical application though its immaturity consider it a work in progress, and a number of implementation-specific details as flagged as areas for further research.

5.1 Algorithm Design

The Immunos implementations discussed are interesting in that they exhibit features not present or not common (to the authors knowledge) in other AIS classification systems. Of particular interest is the competition the technique provides between demarcated groups, (in this case classification labels), at classification time. This is contrary to other AIS systems such as AIRS (Artificial Immune Recognition System) [11] and CLONALG

(Clonal Selection Algorithm) [12,13], (specifically [14,15]) that provide competition at training time. An element of the group-level competition that may prove useful in a new Immunos-level classification technique is the manner in which affinity is affected by antigenicity – that is the size of the group influences the stature of the group indirectly through affinity and directly through an avidity measure.

Another appealing feature of the Immunos technique is the single exposure to training data (a feature also exhibited by AIRS). This is desirable as it causes the system to have predictable training times, which was one of Carter's original design goals. What Immunos lacks is clear and useful generalisation during the formation of the competing groups. Specifically Immunos lacks unit-level competition for free-for-all between all groups that permits the system to effectively stakeout classification boundaries within the data. In the CLONALG and AIRS techniques, (both inspired by the Clonal Selection Theory of acquired immunity); this is achieved through an affinity maturation process of cell proliferation, somatic hypermutation and selection. The addition of this process to Immunos is likely to provide the missing ingredient.

The goal of a new classification technique is to prepare B-cell groups that can identify a particular type of antigen (classification label) that exists in many minor varying forms (data instances), though be able to differentiate between antigen-groups. That is, respond only to one type of antigen, and less so to another class of antigen. Thus, a type of B-cell measure of usefulness can be calculated as how well a B-cell responds to its designated class of antigen compared to the other known classes of antigens. Those B-cells that perform poorly in this respect are negatively selected, and those that perform well in this respect are obviously positively selected. The final population then consists of only those B-cells that are effective at responding more to their antigen-class than any other class. Thus, the final population is expected to be a reduced number of exemplar units compared to the training dataset used.

The end of the training process must see the formation of groups that will compete during classification. To ensure desirable data generalisation properties, it is expected that the size of the prepared groups be less than or equal to the number of data instances of the group observed during training. The systems read-only process of classification will be performed in a similar manner to that of Immunos in that an avidity measure will permit groups to compete to classify an unknown antigen.

Given the general observations and design goals listed, the proposed algorithm will exhibit competition at three levels as follows:

1. Competition during population formation to effectively learn classification boundaries in the training data
2. Competition within each group for affinity assignment during classification
3. Competition between each group using group-specific avidity measures during calculation

The proposed algorithm is called Immunos-99 for lack of a better name.

5.2 Algorithm Specification

In essence, the proposed algorithm can be considered a simplified version of The Clonal Selection Classification Algorithm (CSCA) proposed initially in [15]. The critical point is that this simplification extends to the fact that most user-defined parameters of the technique are removed or fixed, leaving only the essence of the technique. In addition, the structure of the final classifier and the competitive usage of the formed B-cell populations will match those of the Immunos system. Thus, the proposed Immunos-99 is an initial hybrid of core concepts from the CSCA and Immunos-81 algorithms.

5.2.1 Training

The training process is an amalgam of Immunos inspired implementations. The following provides a high-level overview of the principle steps in the training process:

1. Divided data into antigen-groups (by classification labels)
2. Prepare a B-cell population for each group
    a. Create an initial B-cell population for an antigen-group
    b. Loop for a user-defined number of generations
        i. Expose the B-cell population to all known antigens from all groups
        ii. Calculate fitness scorings
        iii. Perform population pruning
        iv. Perform Affinity maturation
            1. Perform fitness-rank based cloning
            2. Perform fitness-rank based mutation
        v. Insert randomly selected Antigens of the same group
3. Perform final pruning for each B-cell population
4. Present the final B-cell populations as the classifier

Training of the system is inspired by elements of both Immunos and CSCA, though if it is accurate to reference the proposed technique as inspired by the clonal selection theory of acquired immunity. First, the algorithm divides the provided antigens into groups demarcated by classification label. Once prepared, a new B-cell population is created from each antigen partition, and each population is prepared in isolation from other prepared populations. This independence or segmentation of the training scheme can be performed in serial (prepare one group at a time), though naturally lends itself to parallelisation.

The initial size of each B-cell population matches the number of antigens in the population's respective antigen-group. The population is then exposed to all antigens for all groups. The population is exposed to one antigen at a time such that an affinity value is calculated for each B-cell to the antigen, and then the B-cell population is sorted in descending order (greatest affinity to least affinity). Note that given that the affinity implementation proposed for Immunos-1 and Immunos2, small affinity scores actually indicate high affinity, thus in this case the population is ordered ascending so that the B-cell with the greatest affinity (smallest value) is at fist position in the ordered list.

The ordered population is iterated, and a rank-based scoring is accumulated within each cell for the antigens-group index (each antigen-group or classification label is allocated a unique index number). Once the population has been exposed to all antigens from all antigen groups, a usefulness score is calculated for each B-cell in the population. This score is taken as the sum rank scores for the B-cells group index divided by the sum rank scores of all other class indices, as follows:

$$fitness = \frac{Correct}{Incorrect}$$

Equation 7 - Calculation for B-cell usefulness or fitness

where *fitness* represents a measure of the B-cells usefulness at recognising same-group antigens. The *correct* score is the sum of rank based scores for antigens of the same group, and the *incorrect* scores is the sum of rank based scores for antigens of different groups to the B-cell.

This calculation for B-cell usefulness is based on that from CSCA, only rather than a histogram of best matches against each known class, the counts are sum rank scores. These scores are taken as follows for a given antigen:

$$score = \frac{i}{N}$$

Equation 8 - Rank-based scoring for each antigen exposure

where N is the total number of B-cells in the ordered population, and i is the rank/index of the i^{th} B-cell in the population starting at a value of one.

Once a usefulness or fitness measure is calculated for each B-cell, the population is pruned back to only those cells that are most useful. This requires a user defined parameter that defines the minimum fitness score (ε) of a B-cell. Unlike the CSCA technique, all B-cells have correct and incorrect scores of varying magnitude. Also unlike CSCA, B-cells cannot change classification label, this means once a B-cell is created for an antigen-group it belongs to that group for life and must compete with all other B-cells of that group for specificity (affinity) to antigens of its classification label, more than to antigens of other classification labels.

Given that each B-cell receives a score for each antigen it is exposed to, based on ranked affinity scores, the final accumulated scores are proportional to the number and diversity of B-cells in the population. Those cells that responded better to self-group antigens have a larger numerator, and will have values > one. Those cells that respond more to non-self antigens have a lager denominator than numerator, and thus have values < one. It is desirable to remove those B-cells with scores less than one, and close to zero, in the range $\varepsilon \in [0,1]$.

Once the B-cell population has been pruned to only those cells of interest, an affinity maturation process is used to improve the remaining population's specificity to its designated antigen-group. This is achieved using rank-based cloning and somatic hypermutation. The population is first ordered by fitness scores in descending order. The number of clones for each B-cell in the list is taken as a rank-proportion of the total number of antigens that belong to the B-cells class from the training set. This means that assuming the pruning process leaves the population untouched, that the population size will double in size for the first affinity maturation step. After the clones are created, they are mutated by the inverse of the B-cells rank ratio, such that small mutations occur to highly ranked B-cells, and larger mutations occur to lower ranked B-cells. This is similar to the approach used in CSCA, only here the ratio is fitness-ranked based and not fitness proportional.

$$r_i = \frac{rank}{S}$$

Equation 9 - B-cell rank ratio calculation

where r_i is the rank ratio of the i^{th} B-cell, rank is the actual index of the B-cell in the ordered sequence $rank \in [1, S]$, and S is the total number of B-cells in the population.

$$numClones_i = \left\lfloor \frac{r_i}{\sum_{j=1}^{S} r_j} \cdot N + 0.5 \right\rfloor$$

Equation 10 - The number of clones created for each B-cell

where r_i is the rank ratio of the i^{th} B-cell and S is the total number of B-cells in the current population, and N is the total number of antigens in the same class (size of antigen-group).

Unlike CSCA, this proposal supports the immune-like behaviour of allowing those cells that are most stimulated more opportunity to proliferate. Further, like CLONALG, it permits hypermutation on progeny in a manner that is inversely proportional to its parent's affinity.

After generated B-cell clones and adding them to the population, a number of randomly generated B-cells are inserted into the population. Rather than generating B-cells from scratch, a sample of randomly selected antigens of the same classification are selected. The total number of new B-cells is equal to the number of B-cells that are deleted during the pruning process. It is expected that inserting antigen-based B-cells may introduce additional diversity into the population in the event that the cloning and pruning process has converged to a limited subset of the problem space. This diversity injection is

provided in the algorithm specification, specifically for the case where the main loop of the algorithm is repeated a number of iterations/generations.

After the above steps have been repeated for each antigen-group for the desired number of generations, the final step in the algorithms procedure is to perform a final pruning step. This step is similar to the fitness-evaluation and pruning step from the main algorithm loop with one specific difference. Each B-cell population is exposed to all antigens, one at a time as occurred previously, only rather than accumulating a rank scoring in all B-cells in the population; only the best matching B-cell receives a score. This causes the stimulated B-cells to form a class-label histogram internally as in the CSCA technique, which is then used for the final pruning phase. The reason for this is that it provides stronger competition and thus better data reduction during the final stage of the classifiers preparation.

The following figure provides a schematic overview of the proposed Immunos-99 algorithm training procedure:

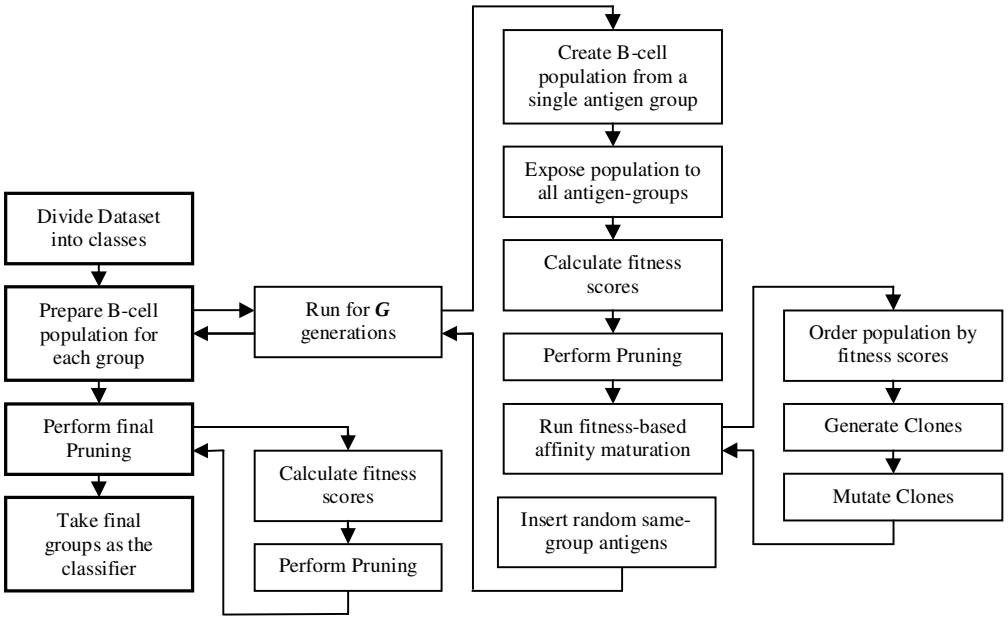

Figure 5 - Schematic overview of the classification algorithms training process

5.2.2 Classification

Once the training procedure has competed, the resulting B-cell populations form the classifier for unknown antigen instances. Classification occurs in the same manner as specified as in the Immunos-1 algorithm. Each B-cell population is exposed to the unknown antigen, and an avidity measure is calculated. The B-cell populations then compete for ownership of the unknown antigen using calculated avidity scores. The population that produces the highest avidity score to the antigen is permitted to apply its classification label.

5.2.3 User Defined Parameters

The system has a minimal number of user parameters, defined as follows:

Algorithm User Defined Parameters

Name	Required / Optional	Description
Total generations (*G*)	Required	The total number of refinement iterations each prepared B-cell population is exposed to. Low values are desired such as one or two. The default value is one.
Random number seed (*r*)	Required	The value with which to seed the random number generator, a parameter common to all algorithms with a stochastic component. Provided for consistency and repeatability of results. It is common to use the system time in milliseconds when these issues are not of concern. Defaults to one.
Minimum fitness threshold (ε)	*Optional*	Used to prune and control the antibody population size. Those antibodies with a fitness allocation ≤ this threshold are deleted from the pool. The parameter has the moderate default value of 0.5.
Seed population percentage (*S*)	*Optional*	The percentage of each antigen-group that is taken as the seed for the B-cell population. Default values include 20% (0.2) to improve speed and data reduction, and 100% (1.0).

Table 4 - Overview of user-defined parameters and prior knowledge for classification algorithm

5.2.4 Statistics

To aid in analysing and understanding the effects of algorithm configurations, a number of training and classifier statistics are proposed, based heavily on statistics proposed in [15].

5.2.4.1 Per-Population Training Statistics

A number of per-population statistics are proposed for each of the prepared B-cell populations. These statistics are designed to be collected and averaged over the number of generations the algorithm is configured for.

1. Population name (classification label)
2. Mean cells pruned per generation
3. Mean population size per generation
4. Mean cell fitness per generation
5. Mean clones per generation
6. Total cells deleted in final pruning

5.2.4.2 Model Statistics

To assess the usefulness and structure of the final classifier produced by the system, a number of classifier-specific statistics are proposed, as follows:

1. Data reduction percentage
2. Total training instances
3. Break down of resource allocations to each group (total cells)

5.3 Preliminary Test Results

The Immunos-99 algorithm was tested on all of the selected datasets in the same manner as the Immnos-1 and Immunos-2 algorithms were tested. Two variants of the algorithm were tested. The first variant seeded (*S*) each of the created B-cell populations with the entire antigen-group (100%). The second variant used a limited percentage (20%) of the antigen-group to seed the created populations. Seeding the population with a reduced number of B-cells is desirable because it requires less initial resources, and permits at least the first generation of the algorithm to execute faster.

A reduced seed population value of 20% was used because it is a relatively low percentage, and because of the observed insensitivity of the initial population size observed in a preliminary analysis of CSCA. It should be noted that the same insensitivity may not apply to the Immunos-81 algorithm, and should be considered an area for further research. Datasets were normalised to unit length as in Immunos-1 and Immunos-2.

During preliminary testing, it was found that a minimum fitness threshold (ε) close to the mean for each B-cell population produced good results. The algorithm implementation was adjusted so that when a minimum fitness threshold value of (-1) is provided the system will use the mean fitness as a cut-off. This is mentioned because some of the test configurations use this feature. Included in this adjustment is an upper-limit of one. Given that values are in the range $\varepsilon \in [0,1]$, a mean value > one always taken to one.

Finally, it should be noted that algorithm configurations were found using a simple ad-hoc, trial-and-error method. As such, the results must be considered preliminary and non-optimal. Further, it should be noted that during initial testing, the system was observed to have a low-sensitivity to the values specified, producing similar results across a range of parameter configurations.

5.3.1 Test Results – Seed with entire group

Configuration

Dataset	Minimum Fitness Threshold (ε)	Total Generations (G)
Cleveland Heart Disease (CHD)	-1	2
Iris Plants (IP)	1	1
Sonar (S)	0.6	2
Wisconsin Breast Cancer (WBC)	0.5	1

Table 5 – Algorithm configuration for each selected dataset

Immunos-99

set	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Mean	Stdev.	Min	Max
	81.518%	81.188%	81.848%	80.858%	81.188%	81.188%	81.188%	81.518%	81.518%	81.188%	**81.32%**	0.264	80.858%	81.848%
	96.667%	97.333%	97.333%	97.333%	97.333%	97.333%	97.333%	97.333%	98%	97.333%	**97.333%**	0.298	96.667%	98%
	68.269%	68.269%	67.788%	67.788%	69.231%	69.231%	69.231%	67.788%	70.192%	67.308%	**68.51%**	0.867	67.308%	70.192%
	81.974%	82.117%	81.545%	81.688%	83.405%	81.974%	82.26%	83.262%	81.688%	81.688%	**82.16%**	0.624	81.545%	83.405%

Table 6 - Immnos-99 classification accuracy results for selected datasets

Immunos-99

Dataset	Mean	Stdev.
CHD	**15.816%**	0.72
IP	**6.8%**	1.166
S	**20.781%**	1.128
WBC	**45.888%**	0.906

Table 7 - Immnos-99 data reduction results for selected datasets

5.3.2 Test Results – Seed with 20% of group

Configuration

Dataset	Minimum Fitness Threshold (ε)	Total Generations (G)
Cleveland Heart Disease (CHD)	-1	1
Iris Plants (IP)	-1	2
Sonar (S)	0.35	1
Wisconsin Breast Cancer (WBC)	-1	1

Table 8 – Algorithm configuration for each selected dataset

Immunos-99

set	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Mean	Stdev.	Min	Max
	81.518%	81.518%	82.508%	82.178%	82.508%	80.528%	81.848%	81.518%	80.858%	80.858%	**81.584%**	0.657	80.528%	82.508%
	96%	96.667%	95.333%	88.667%	96.667%	97.333%	97.333%	96.667%	97.333%	98%	**96%**	2.547	88.667%	98%
	55.769%	62.019%	62.019%	62.019%	62.019%	62.019%	62.019%	64.904%	62.981%	60.096%	**61.587%**	2.239	55.769%	64.904%
	84.835%	84.692%	83.262%	83.262%	84.692%	84.406%	83.691%	83.548%	84.549%	84.406%	**84.134%**	0.591	83.262%	84.835%

Table 9 - Immnos-99 classification accuracy results for selected datasets

Immunos-99

Dataset	Mean	Stdev.
CHD	**68.962%**	2.216
IP	**27.5%**	4.478
S	**79.271%**	1.206
WBC	**78.762%**	1.01

Table 10 - Immnos-99 data reduction results for selected datasets

5.4 Discussion and Further Research

The test results for Immunos-99 were interesting both in terms of the accuracy provided and the concepts tested in the algorithms implementation. In terms of classification accuracy, Immunos-99 was capable of out-performing Immunos-1 and Immunos-2 for all datasets except the WBC. The gains achieved by the algorithm over the previous two implementations were not significant averaging one or less than one points in accuracy.

When Immunos-99 was seeded with 20% of its antigen-group, it showed minor increases in accuracy for the CHD and WBC datasets compared to being seed with the entire antigen-group. This is interesting given that both datasets are medical-based, and Carter originally intended the Immunos-81 algorithm for medical datasets (likely just a coincidence). The most interesting difference between the two variants of the Immunos-99 algorithm is by far the difference in data reduction. For the datasets tested, a difference of 20% to 58% difference in data reduction was observed in favour of the 20% seeded version. On all but the Iris plants dataset, the second variant of the algorithm was achieved a reduction in data above close to 70% and 80%, compared to the best achieved by the first variant of 45%.

The poor lower data reduction observed for the IP datasets for the second variant is expected to be caused by the fact that the algorithm was run for two generations, as opposed to the one generation runs for all other dataset. This and other preliminary observations lead to the hypothesis that the more generations the algorithm is executed for the less data reduction is obtained due to a specialisation to the training dataset. It is expected that when analysed across a number of datasets that a performance graph for different numbers of generations would reveal an increased accuracy trade-off at the expense of data reduction. Further, it is speculated that a graph will reveal a point of degradation where the system becomes too specialised to the dataset and is unable to effectively generalise. These behaviours are based on similar behaviours commonly observed of artificial neural networks.

The mean fitness as a minimum fitness cut-off value as implemented during the algorithm testing was shown to be a useful parameter heuristic. The results showed that 1/3 of runs and 3/4 of runs for the first and second algorithm variants used this feature. During algorithm runs, it was observed that when set to a low value, the system performed little to no pruning during the main loop, leaving all pruning to the final pruning step. The usefulness and sensitivity of this parameter requires further investigation. In addition, the fitness-allocation methods (rank and best matching unit) require a detailed analysis to assess their impact on the pruning process both during the main loop and after the main loop of the algorithm.

For all test results, the number of generations was low, typically having the value of one. This matches the expectation presented by the original Immnos-81 technique that is capable of preparing the system using a single pass over the training data. The B-cell proliferation and hypermutation processes of the algorithm require further attention and development. During initial testing, it was found that similar results can be obtained when

reversing the rank order for number of clone, and mutation range allocations. This may be a feature of the algorithms in-sensitivity to configuration, though is more likely a result of selecting an unsuitable B-cell proliferation scheme.

Finally, the use of a population-specific minimum-fitness cut-off (mean) raises the question of the usefulness of other population-specific parameters. Given the independence in training, each population naturally lends to the idea of independence of configuration. It may be useful to use population specific stop conditions, number of generations and even algorithm structure such as fitness allocation and pruning methods. This is another area for further research.

6. Conclusions

This work has shown that the Immunos-81 system is not sufficiently described to replicate. Given this fact, it was shown that it is possible to extract the general principles of the technique into what was termed generalised Immunos-81. From these principles, two so called “interpreted implementations” of the technique were tested indicating that there may be some interesting and/or useful elements from the proposed system for immune-inspired classification systems.

Extending upon the general principles of the systems and the lessons learned in implementing the Immunos-1 and Immunos-2, the Immunos-99 algorithm was designed, implemented and tested. The algorithm showed that it is possible to integrate desirable features from Immunos-81 and cell iterative affinity maturation through affinity proportionate proliferation and hypermutation. Although the resulting classification system out-performed its parents (Immunos-1 and Immunos-2), it still has a lot of room for improvement to be competitive will other widely used classification systems described on [9,10].

7. Appendix – WEKA Algorithm Implementation

WEKA is a machine learning workbench [16] written in the Java programming language. It provides a large number of common classification, clustering, attribute selection algorithms as well as visualisation tools, algorithm test schemes and data filtering tools. WEKA provides a number of interfaces for making use of the tools and algorithms provided such as a command line interface, a data exploration interface, an algorithm test and comparison interface, a workflow interface, and finally a programmer level application programming interface for integrating WEKA functionalities into standalone allocations. WEKA has been made open source, allowing academics and industry to extend the platform by adding algorithm and tool plug-ins for the platform.

As apart of this work investigating the Immunos-81 algorithm, a number of Immunos-based algorithms were implemented for the WEKA platform. These algorithm implementations are located in the package "weka.classifiers.immune" and are accessed as such in the WEKA user interfaces from the algorithm selection drop-down.

7.1 Immunos-1 & Immunos-2

The Immunos-1 and Immunos2 algorithm implementations are simplistic and provide no user defined parameters. Both algorithm automatically normalise the training set before preparing the classification model. No statistics are provided for these algorithm implementations.

7.2 Immunos-99

The Immunos-99 algorithm is implemented as defined in Section 5. The technique has four user defined parameters and like Immunos-1 and Immunos-2, normalises the training dataset automatically before presentation to the algorithm. The system provides all the statistics described in Section 5.2.4 in an effort to analyse the techniques performance in response to adjusted user parameters. The following figure provide an indication of the techniques configuration in the WEKA Explore graphical user interface:

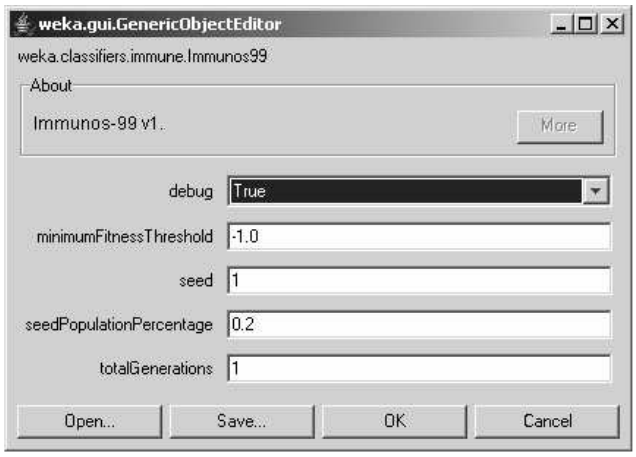

Figure 6 - Immunos-99 configuration panel in WEKA GUI

The following provides example statistics from applying the technique to the Iris Plants dataset with default parameters:

```
Immunos-99 v1.0.

 - Training Summary -
Group name: Iris-setosa
Cells pruned per generation:......7 (0)
Population size per generation:...63 (0)
Cell fitness per generation:......0.714 (0)
Cloned cells per generation:......50 (0)
Cells deleted in final prune:.....47

Group name: Iris-versicolor
Cells pruned per generation:......4 (0)
Population size per generation:...58 (0)
Cell fitness per generation:......0.497 (0)
Cloned cells per generation:......50 (0)
Cells deleted in final prune:.....48

Group name: Iris-virginica
Cells pruned per generation:......5 (0)
Population size per generation:...59 (0)
Cell fitness per generation:......0.676 (0)
Cloned cells per generation:......50 (0)
Cells deleted in final prune:.....46


 - Classifier Summary -
Data reduction percentage:...74%
Total training instances:....150
Total cells:................39

 - Classifier Memory Cells -
Iris-setosa: 16
Iris-versicolor: 10
Iris-virginica: 13
```

Figure 7 - Example algorithm statistics for Immunos-99 on the Iris Plants dataset

7.3 Algorithm Usage

As mentioned, the algorithm implementations can be used directly from the WEKA Explorer, WEKA Work Flow, and WEKA Experimenter interfaces. For the algorithm implementations to be recognised by WEKA, the ImmunosWeka.jar file must be in the Java class path. The implementation was prepared with Java 5.0 (1.5), and thus the

installed Java Runtime Environment (JRE) must be also be this version. Finally, the version of WEKA that the algorithm was prepared for and tested with is 3.4.3.

The following provides examples of using the three algorithm implementations from the command line on the Iris Plants dataset with 10-fold cross-validation.

```
java –cp weka.jar;ImmunosWeka.jar
weka.classifiers.immune.Immunos1 -t data/iris.arff

java –cp weka.jar;ImmunosWeka.jar
weka.classifiers.immune.Immunos2 -t data/iris.arff

java –cp weka.jar;ImmunosWeka.jar
weka.classifiers.immune.Immunos99 –s 1 –S 0.2 –G 1 –E -1 -t
data/iris.arff
```

Figure 8 - Shows examples of executing the two WEKA implementations from the command line

The following code sample provides an example application of using the Immunos-99 algorithm in standalone mode. The program loads the Iris Plants dataset and performs a 10-fold cross-validation test.

```
public class SimpleCSCAUsage
{
     public static void main(String[] args)
     {
          try
          {
            // prepare dataset
            Instances dataset = new Instances(
              new FileReader("data/iris.arff"));
            dataset.setClassIndex(dataset.numAttributes()-
1);
            Immunos99 algorithm = new Immunos99();
            // evaulate
            Evaluation evaluation = new Evaluation(dataset);
            evaluation.crossValidateModel(algorithm,
              dataset, 10, new Random(1));
            // print algorithm details
            System.out.println(algorithm.toString());
            // print stats
            System.out.println(
              evaluation.toSummaryString());
          }
          catch(Exception e)
          {
               e.printStackTrace();
```

```
            }
        }
}
```

Figure 9 - Shows Java code implementation of an application using Immunos-99 to classify the Iris plants dataset

8. Bibliography

[1] Jerome H. Carter, The immune system as a model for classification and pattern recognition *Journal of the American Informatics Association*, vol. 7, 2000.

[2] J Timmis, T Knight, L N De Castro, and and E Hart. An Overview of Artificial Immune Systems. In: *Computation in Cells and Tissues: Perspectives and Tools for Thought*, Anonymous Springer, 2004.pp. 51-86.

[3] Emma Hart, Immunology as a Metaphor for Computational Information Processing: Fact of Fiction 2002. University of Edinburgh.

[4] Jamie Twycross, An Immune System Approach to Document Classification 2002. University of Sussex.

[5] Donald Goodman, Lois Boggess, and Andrew Watkins, "An Investigation into the Source of Power for AIRS, an Artificial Immune Classification System," *Proceedings of the International Joint Conference on Neural Networks (IJCNN'03),* pp. 1678-1683, 2003.

[6] Andrew Watkins, Xintong Bi, and Amit Phadke, "Parallelizing an Immune-Inspired Algorithm for Efficient Pattern Recognition," *Intelligent Engineering Systems through Artificial Neural Networks: Smart Engineering System Design: Neural Networks,* pp. 225-230, 2003.

[7] Andrew Watkins and Jon Timmis, "Exploiting Parallelism Inherent in AIRS, an Artificial Immune Classifier," *Proceedings of the 3rd International Conference on Artificial Immune Systems (ICARIS2004),* Catania, Sicily, Italy, pp. 427-438, 2004.

[8] C. L. Blake and C. J. Merz. UCI Repository of Machine Learning Databases. [Online] http://www.ics.uci.edu/~mlearn/MLRepository.html . 1998. University of California, Irvine, Dept. of Information and Computer Sciences.

[9] Wlodzislaw Duch. Datasets used for classification: comparison of results. [Online] http://www.phys.uni.torun.pl/kmk/projects/datasets.html . 2002. Computational Intelligence Laboratory, Department of Informatics, Nicholaus Copernicus University, Torun, Poland.

[10] Wlodzislaw Duch. Logical rules extracted from data. [Online] http://www.phys.uni.torun.pl/kmk/projects/rules.html . 2002. Computational Intelligence Laboratory, Department of Informatics, Nicholaus Copernicus University, Torun, Poland.

[11] Andrew Watkins, Jon Timmis, and Lois Boggess, Artificial Immune Recognition System (AIRS): An Immune-Inspired Supervised Learning Algorithm *Genetic Programming and Evolvable Machines*, vol. 5, pp. 291-317, Sep, 2004.

[12] Leandro N. de Castro and Fernando J. Von Zuben, "The Clonal Selection Algorithm with Engineering Applications," *GECCO 2000, Workshop on Artificial Immune Systems and Their Applications,* Las Vegas, USA, pp. 36-37, 2000.

[13] Leandro N. de Castro and Fernando J. Von Zuben, Learning and Optimization Using the Clonal Selection Principle *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, vol. 6, pp. 239-251, 2002.

[14] Jennifer White and Simon M. Garrett, "Improved Pattern Recognition with Artificial Clonal Selection," *ICARIS-2003,* Edinburgh, pp. 181-193, 2003.

[15] Jason Brownlee, Centre for Intelligent Systems and Complex Processes (CISCP), Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology, Victoria, Australia, Technical Report ID: 2-01, Jan 2005.

[16] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools with Java implementations*, San Francisco: Morgan Kaufmann, 2000.