Natural Immune Systems for Developing Novel Machine Learning Algorithms

A thesis submitted to

Auckland University of Technology (AUT)

in fulfilment of the requirements for the degree of

Doctor of Philosophy (PhD)

Waseem Ahmad

2012

School of Computing and Mathematical Sciences

Primary Supervisor: Prof. Ajit Narayanan

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

"I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another author (expect where explicitly defined in the acknowledgements), nor material which to a substantial extension has been submitted to the award of any other degree or diploma of a university or other institute of higher learning."

Signed: _____

Acknowledgements

There are a number of people who without their support I would never have made it this far.

First and foremost, I am deeply grateful to my supervisors Prof. Ajit Narayanan and Dr. Russel Pears for their invaluable suggestions, support and encouragement during the course of this research. This thesis would not have been possible without their guidance and being their student has been a great honor for me. The work presented here in this thesis owes much to their enthusiasm and careful supervision.

I am grateful to numerous staff and colleagues in the PhD Lab at the AUT University for their friendship and advice, as well as their moral and social support. My special thanks go to Silu Cao, for her enduring friendship and support during the course of my research in New Zealand.

I would like also to thank Mr. Hassan Nisar, a scholar and a columnist, whose writings and ideology have always been a great source of inspiration for me.

And, finally, I am heartily grateful to my family and especially my parents, who were always there to support and provide me with resources and the means with which to complete this thesis.

Abstract

This thesis presents novel multi-layered natural immune system (NIS) inspired algorithms in the domain of machine learning. The exploration of biological metaphors for developing novel learning algorithms for computation is not new. The contribution of artificial neural networks inspired by the human brain, genetic algorithms inspired by modern-day genetics, and ant colony algorithms inspired by swarm intelligence is well recognized. A relatively new addition is the artificial immune system (AIS) that is inspired by an NIS, where micro-level processes and metaphors of NIS are used to develop novel computing algorithms.

In recent years, two main AIS algorithms, namely, CLONALG and aiNet, have been developed based on the principles of 'clonal selection' and 'immune network theory' respectively. Both of these algorithms can be regarded as data reduction processes with limited learning capability. The role of evolutionary computing in these algorithms is restricted to the hypermutation of antibodies in response to pathogens, however. In this thesis, the role of evolutionary computing in AIS is broadened significantly to include the evolution of learning parameters and an active role of memory cells, as well as a population-based approach to AIS. A novel AIS algorithm inspired by the humoral mediated immune response (HIR) triggered by the adaptive immune system is presented. In humoral immune response, antibodies with hypermutation are secreted to mount an appropriate immune response to pathogens. This thesis introduces the concept of a layered architecture, where pathogens are identified and captured using different layers of cells such as antibodies, memory cells, and B-cells.

To date, the focus of AIS researchers has been on developing novel AIS algorithms using various NIS metaphors and processes. Relatively little research has focused on integrating AIS models with other nature-inspired techniques to achieve effective and improved learning capabilities. In this thesis, we propose a novel methodology inspired by the vaccination process in an NIS, where information contained in the memory of an AIS is used to prime learning in a hybrid architecture. We demonstrate the effectiveness of this hybrid architecture and explore future directions of AIS.

Chapter 1

Introduction

1.1 Research Questions

We can express the contribution of this thesis by proposing the following two research questions:

Q1: *Is it possible to develop an intelligent and biologically plausible learning algorithm inspired by the processes and metaphors of the natural immune system (NIS), informed by the latest scientific research?*

Q2: *Is it possible to incorporate NIS concepts and metaphors with other well-established nature-inspired techniques to achieve effective and improved learning capabilities?*

The effectiveness of a learning algorithm can be measured using quantitative techniques based on internal or external evaluation criteria [1]. In this thesis, effectiveness is measured by comparing the results obtained by a learning algorithm against true results/labels (external criteria). Improvement on the other hand, in learning systems is measured by (1) comparing how fast an algorithm can learn (in terms of learning cycles), and (2) comparing results of final learned model against other benchmarked learning models.

The rest of this chapter provides the motivation, scope and organization of this thesis and backgrounds the two research questions.

1.2 Motivation

Nature, over millions of years has found innovative, robust and effective methods for dealing with the challenges faced by living creatures every day. The primary mechanism for these methods is 'neo-Darwinism', which is evolution by natural selection underpinned by modern genetics. This has led to the emergence of a relatively recent area of computing called 'evolutionary computing', which refers to a collection of nature-inspired techniques for solving hard problems in computer science. For instance, the contributions of genetic algorithms [2, 3], simulated annealing [4, 5], ant colony optimization (ACO) [6, 7], and particle swarm optimization (PSO) [8] algorithms in different disciplines are now well recognized. One feature of evolutionary computing and nature-inspired techniques so far is their focus on evolutionary methods at the level of an individual organism or in populations of organisms. Relatively less explored or understood is how the latest advances in our understanding of the genetic mechanisms underlying evolution can be used to derive new algorithmic solutions at micro-organismic level. The goal of this thesis is to explore how nature-inspired models informed by our latest understanding at the genetic or bimolecular level can be used to develop novel, nature-inspired algorithms. Our primary focus of interest will be the NIS and the inspiration it provides to computational techniques ('Artificial Immune Systems - AIS') [9-11].

There is a rapidly growing interest in AIS approaches to machine learning. Of particular interest is the way the human body responds to diseases and new pathogens as well as adapting to remain immune for long periods after a disease has been combated. Immune system processes consist of two phases: recognition of invaders, and response. It has been established that the NIS can adequately distinguish between threat and non-threat at a basic level. Also of interest is the way that the NIS can identify a self-cell (which is not to be reacted to) but which has been subsequently damaged in some way and might present a threat to the body (and which must be reacted to). In other words, the NIS is dynamic in that it can re-structure (re-classify or re-cluster) in the light of new information so that it provides protection against not only outside invaders but also inside dangers. All these NIS concepts, if carefully and systematically used, could

confer great benefit in the area of machine learning. However, it is currently uncertain as to how ‘deep’ one should go into the biological background and still have effective algorithms (Q1 and Q2). That is, the deeper one goes into the biology, the more complex the mechanisms become from a computational perspective. But these complexities could give rise to new algorithms or the application of current techniques in a novel way, as well as improved methods for, in our case, learning.

AISs fall in the group of biologically inspired computing. Biologically inspired computing (or bio-inspired computing) is a sub-routine of artificial intelligence (AI), where metaphors derived from biology are used to improve on existing computational designs and software. This is different from computational biology, where computer software and hardware are used to simulate real biological systems. Learning is one of the common and most important features of all bio-inspired or nature-inspired computing [12].

The source of nature and biological-inspired computing techniques for learning (supervised and unsupervised learning) has its roots in our early and macro-level understanding of biological processes. The contribution of artificial neural networks (ANNs) [13] inspired by our knowledge of the human brain at the level of the neuron and neuronal interconnectivity is now well recognized as providing a number of effective supervised and unsupervised algorithms for classification and clustering respectively [13-17]. ANNs have been successfully used in the field of pattern recognition and other machine learning domains [18-20]. Another example of nature-inspired computational techniques is genetic algorithms (GAs), inspired by neo-Darwinian evolutionary principles [21]. Mutation, crossover, natural selection, and reproduction are some of the most important features of GA. With GAs, a population of solutions are evolved using evolutionary principles such as mutation, crossover and natural selection based on some fitness criterion. GAs have also been successfully used in optimization, supervised, and unsupervised learning tasks. Again, GAs are largely inspired by our understanding of genes and chromosomes at a macro-level. In recent years the contributions of ACO and PSO algorithms [22-25] have been well recognized in the field of data mining. ACO and PSO are inspired by ‘super-macro’ phenomena emerging at the level of populations (colonies and swarms, for instance) rather than at the level of individuals in a population.

By the start of this century, there had been a paradigm shift in computing researchers from macro-level processes to micro-level processes for their inspirations to build new computing techniques. The emergence of DNA computing [26, 27], quantum computing [28, 29] and NIS-inspired computing [30-33] are a few examples of this shift in focus. The main hypothesis, which most researchers work on, is that the deeper into the natural systems they go, the more improved their algorithms become. Here, improvement can be defined in terms of computational speed, and effectiveness in terms of quality of results or any novel information contributing to existing knowledge. In this thesis, we will explore NIS concepts, processes and metaphors to develop improved novel learning algorithms in the field of machine learning. The task will be to go at least one-level 'deeper' into the biology of immune systems than so far achieved with AISs to see what improvements are possible, if any.

1.3 Scope

1.3.1 Machine Learning

Mitchell provided a widely quoted definition of machine learning: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." [34]

Machine learning is a sub-field of artificial intelligence (AI). The field of machine learning studies the design of computer programs, not only to learn from existing patterns to make better decisions under current circumstances, but also help in making better future decisions by generalizing current experiences. For example, in the case of a cancer patient data record, a machine learning algorithm must be able to describe relationships between disease and non-disease, and various symptoms. In addition it should be able to predict the presence or absence of cancer in future patients effectively. The rationale for using machine learning techniques over conventional statistical techniques is as follows [35]:

1. Some tasks are not well defined and can only be acquired through examples.
2. Machine learning helps to extract hidden patterns from large amount of data, which might not be apparent to the naked eye.

3. Machine learning characteristics such as adaptation and learning, can handle non-deterministic environment in a much more efficient manner.

4. Machine learning algorithms can adapt in dynamic environments.

Machine learning techniques can be classified into three groups: supervised learning, unsupervised learning, and reinforcement learning [12].

Supervised Learning: In supervised learning, a model is built based on mapping input patterns to desired output. Data consists of sets of features or variables as well as output (desired) variables. The task of any supervised learner is typically to build a model that both fits a number of training examples (training data) and predicts the class into which new or unseen data falls, where the class of the instance may not be known (test data). Over the past few decades many supervised learning algorithms have been proposed and published, including classifiers based on information gain (e.g. decision trees and rule-based classifiers) [36, 37], probabilistic and statistical learning techniques (e.g. Bayesian networks, Naïve Bayes classifiers, linear discriminant analysis) [38, 39], support vector machines [40], and neural networks [20, 41].

Unsupervised Learning: The aim of unsupervised learning is to find groupings in data when class labels (desired output) or any other related information is not available. Therefore, unsupervised learning algorithms seek to group similar data into clusters (groups) so that all data instances within a group have maximum similarity, while instances across different clusters have a high degree of dissimilarity. Over the years, many different methods of unsupervised learning have been developed ranging from the simple K-means algorithm [42] to more sophisticated methods such as expectation maximization [43], the neural network-based self-organizing map [14], and spectral clustering [44]. It has been applied in a number of different ways in data mining and analysis, including visualization [45], dimensionality reduction [46], and as a pre-processing tool to other data mining methods such as classification and association rule mining [47].

Reinforcement Learning: These algorithms do not depend on explicit supervision or optimization of an objective function, as in the case of supervised or unsupervised learning approaches. Reinforcement learning algorithms interact directly with the environment. It is the environment that provides the feedback in the form of rewards or

punishments that guide the learning algorithms. Reinforcement learning is the process by which an agent learns and adapts its behavior through trial-and-error interactions with a dynamic environment [48]. These algorithms tend to be slow in finding solutions. However, they perform an exhaustive search, which helps to explore more search space to find local optimal solutions. A survey of reinforcement learning methods can be found in [48].

In this thesis, our focus is to develop novel machine learning algorithms in the domains of supervised and unsupervised learning by incorporating inspirations from natural systems (and more specifically the NIS).

1.3.2 Natural Systems

The main motivation for using natural systems as an inspiration for computation is due to the fact that natural systems have complex, intelligent and non-linear behaviors, which arise from simple, decentralized and linear processes. In our everyday life, we have to devise various strategies to survive. These strategies can mainly be classified into two basic behaviors or expressions: 'co-operation' and 'competition' (or combination of both). In neo-Darwinism, an evolutionary strategy such as 'survival of the fittest' is adopted where, based on some environmental fitness function, individuals are evaluated and the weaker individuals die out due to natural selection. This characteristic in natural systems gives rise to competition among individuals. On the other hand, natural systems also encourage co-operation. The formulation of multi-cellular organisms from single-cell organisms is an example of co-operation that leads to the emergence of distributed learning systems. Another example of co-operation is the human nervous system that consists of neurons which form a complex network of interconnected cells and various signals are inhibited or exhibited for information processing. It is these characteristics of co-operation and competition among various cells and organs in natural systems that give rise to 'homeostasis'. Homeostasis is a healthy and stable state that any natural system tends to stay in, at all times [33]. This homeostatic state is acquired at both intra-system and inter-system levels through the interaction of cells in a system and cells across various systems, respectively. The NIS achieves an internal state of homeostasis by the interaction of its various organs and processes. On the other hand, at the organism level, different systems, such as immune

system, nervous system, or respiratory and circulatory systems also maintain a state of homeostasis state, to keep us alive.

Nature exemplifies supervised learning (classification) and unsupervised learning (clustering) in effective ways. Classification is dependent on some objectively measured value (class value) that can be used to determine the importance of attributes for classifying samples, whereas clustering depends only on shared values or patterns of similarity among the samples themselves. Some examples of natural clustering and classification are given below:

Flock of birds: A flock of birds in flight changes shape, orientation, and density, as well as divides and regroups. Key features of a flock are a relatively clearly defined boundary to the flock or sub-flocks in flight, roughly equal distances between all members of the flock/sub-flock, and no clear leader or centralized control of the flock/sub-flock. These features emerge at the macro-level (organism or population levels) [49] due to the interaction between an individual in the flock and its nearby neighbors. In other words, individuals belonging to a flock or sub-flock share properties between members of the flock. If there is an objective value independent of the attributes of the flock, such as one flock heading north and another heading south, this value can be used to generate class labels for the two flocks. If there is no independent value, one flock can only be distinguished from another flock because of shared properties in one flock being sufficiently different from the shared properties of the other flock, according to some similarity/dissimilarity function.

Formation of ant cemetery: Another example is the formation of an ant cemetery. Some ant species have the capacity to gather their corpses at the same location (a form of clustering) without central knowledge of the location of the cemetery [50]. This formation of cemetery by ants is not a collective behavior of a group since each ant works independently without the knowledge of the positioning of other ants.

Natural immune system (NIS): The NIS at an abstract level can differentiate between self and non-self (harmless and harmful). Therefore, this is a perfect case of two-class classification/clustering problem. This process happens at the micro biology level (within and between cell level) [9]. The NIS consists of two subsystems, the innate immune system and the adaptive immune system [51]. The innate immune system is what inherits and it stays fairly constant throughout our life span. On the other hand, the

adaptive immune system is dynamic and it keeps evolving as new pathogens are encountered.

The NIS helps to protect us from harmful bacteria and viruses. It also provides us with protection from cells, which were initially harmless but due to infections or alterations become harmful to the body (e.g. cancer cells). Therefore, our immune system not only protects us from outside invaders but also from inside invaders. A lot of research has been carried out in last couple of decades that has significantly helped our understanding of the NIS. For computation researchers, the self-evolving, self-sustaining, and self-organizing behaviors of the NIS are of great interest. Moreover, it's embedded characteristics such as learning (through adaptation) and memory of past encountered pathogens can be helpful in building robust learning algorithms. Over the years, many theories have been proposed regarding the functioning of the NIS. Two immunology theories, namely, immune network theory and clonal selection theory have attracted great interest in the computational community. The immune network theory proposed by Jerne [52] suggests that cells in an immune system operate by interaction with each other as well with pathogens to form a network of cells and these networks act as a memory of network. On the other hand, clonal selection theory states that cells in the adaptive immune system that are stimulated by pathogens are transformed into memory cells, which is a way of creating memory in an immune system. Both these theories explain the formation of memory of previously encountered pathogens in the adaptive immune system. This thesis looks closely at both of these theories to derive inspiration and also looks into other immune system processes in order to develop a novel immune system-inspired learning algorithm.

1.3.3 NIS for Computation

The main objective of this thesis is to apply NIS concepts and metaphors in the domain of supervised and unsupervised learning. One interesting question that can be asked is: Why use the immune system as inspiration for supervised or unsupervised learning when thousands of algorithms already exist in the literature to solve these problems? The answer lies in the No Free Lunch Theorem [53]. This states that there cannot exist an algorithm that, on average, is superior to any other algorithm. Also, according to Jain [54], each clustering algorithm is designed to solve a specific task and therefore these algorithms implicitly or explicitly impose structural constraints on the data. One of the

examples of such constraints is preference bias [55]. Different similarity measures such as Euclidean distance, Cosine distance and Hamming distance are examples of preference bias. Representation bias is another example of such constraints. This bias arises due to the choice of data representation (such as continuous, binary or discrete data) [34]. The hypothesis of 'no best algorithm' is also explained by the impossibility theorem [56], which states that, given the three basic conditions (axioms) of data clustering, namely scale invariance, consistency and richness, there is no clustering algorithm that can satisfy all these conditions simultaneously. Therefore, all existing algorithms make some trade-off on these conditions. In conclusion, there is always a need for a new algorithm that can explore the data in novel ways. In this thesis, we will try to develop novel learning algorithms (supervised and unsupervised learning algorithms) using inspirations from the NIS.

In 1782 Jenner discovered that a smallpox viral attack can be prevented if people are injected with a small amount of the cowpox virus – a weaker form of the smallpox virus found in cattle. This is a form of vaccination, where a weaker form of pathogen is introduced in living organisms to immunize the body from stronger pathogens of the same kind. This process of vaccination highlights important features of the NIS, namely adaptation and generalization. Firstly, the NIS can adapt when new viruses are encountered. Secondly, once it adapts itself to a weaker form of virus or pathogen, the NIS can generalize to stronger viruses and pathogens as well. It is these adaptation and generalization characteristics of the NIS that have been the focus of computing researchers in recent years. Apart from above mentioned characteristics, NISs also demonstrate the following capabilities [57]:

1. **Learning:** An NIS continuously learns and adapts from the pathogens to trigger an appropriate immune response.

2. **Diversity:** An NIS consists of various cells and organs, which help it to function and mount an immune response when pathogens are encountered.

3. **Specialization/Generalization:** An NIS has capabilities of specialization as well as generalization. Specialization is achieved through the presence of memory cells whereas generalization is acquired through the generation of antibodies.

4. **Memory:** Memory of previously encountered viruses and pathogens is kept in the form of memory cells, so that if the same pathogen attacks the immune system in the future, it can trigger a fast and more effective response.

5. **Multi-layered:** An NIS has various layers. Innate and adaptive immune systems are the two main layers in an NIS. Human skin and various secretions also provide supplementary layers and form an integral part of an NIS.

6. **Decentralized process:** An NIS is decentralized in nature, meaning it does not have any central control. Also, it is distributed all over the body and consists of various cells and organs.

7. **Noise tolerance:** An NIS is tolerant to noise, and a perfect match between pathogen and immune cell receptors is not required to trigger an immune response.

8. **Dynamic system:** An NIS is constantly under attack by new pathogens and therefore it is constantly changing and adapting to new pathogens. As pathogens and viruses are evolving all the time, an NIS has to be dynamic to trigger an appropriate immune response.

1.4 Thesis Details

Returning to the two research questions (listed earlier in section 1.1), we have now provided the basic motivation underlying these research questions. That is, the main aim of this thesis is to go at least one level deeper than previous AIS algorithms, into the biological mechanisms that support NISs and to inform our AIS algorithms with information and processes found at this deeper level. In the final sections of the thesis, we will evaluate whether going deeper has provided any benefit. The following sub-sections will discuss the contribution of this thesis as well as outline the organization of the remainder of the thesis.

1.4.1 Thesis Contribution

The contribution of this thesis to existing knowledge can be summarized as follows:

1. An extensive literature review of existing AIS work is presented and several drawbacks of existing AIS approaches are highlighted in chapter 2.

2. A multi-layered, immune-inspired unsupervised clustering algorithm we call the Humoral-mediated Artificial Immune System (HAIS) is proposed in chapter 3, based on the latest biological knowledge. Experiments on simulated and real world datasets conclude that the proposed algorithm is efficient at finding natural grouping (clusters) and outliers in the data simultaneously.

3. A detailed analysis of the processes and parameters involved in HAIS algorithm is presented in chapters 4 and 5. We demonstrate that our proposed HAIS algorithm has learning, data summarization, generalization, and adaptive memory capabilities.

4. A multi-layered immune-inspired supervised learning algorithm is proposed in chapter 6. The experiments conducted using the supervised HAIS algorithm demonstrates that it has efficient data summarization and generalization capabilities using NIS metaphors.

5. A hybrid architecture is presented in chapter 7, where the integration of immune system concepts with artificial neural networks is proposed to achieve a robust and improved supervised learning model.

1.4.2 Thesis Organization

The rest of the thesis is organized into the following seven chapters:

Chapter 2: An introduction to the NIS is provided in this chapter. The NIS is a very broad area; therefore, we only discuss NIS concepts that are related to the scope of this thesis. An extensive literature review of immune system-inspired approaches is also presented, with special focus on AIS applied to supervised and unsupervised learning domains. This chapter concludes with highlighting some drawbacks of existing approaches, which form the basis of our novel immune system-inspired learning algorithm.

Chapter 3: Based on the limitations and drawbacks discussed in the previous chapter, a novel AIS clustering algorithm, which we call the Humoral-mediated Artificial Immune System (HAIS), is proposed. The HAIS is inspired by the humoral mediated immune response (HIR) triggered in the adaptive immune system. The HAIS derives its inspirations from the way an adaptive immune system actively produces antibodies once

encountered by pathogens. One advantage of the HAIS algorithm is its capability of finding natural groupings and outliers (clusters) in the data simultaneously. The performance of the novel HAIS algorithm is evaluated against well-known clustering methods on simulated and benchmark real world datasets.

Chapter 4: After looking closely into the behavior of HAIS and evaluating its parameters in previous chapter, an extension of HAIS, a population-based HAIS, is proposed. In the population-based HAIS, the standard HAIS algorithm coupled with genetic algorithm are used to evolve the population of clustering solutions to achieve better unsupervised clustering results. Memory cells form the basis of incremental learning, i.e. the transfer of knowledge from the current generation to the next generations.

Chapter 5: This chapter looks into the main components of HAIS algorithm and evaluates each component separately. A novel methodology based on a variation of the HAIS algorithm is designed to evaluate each of the components to observe their contribution to the proposed HAIS algorithm to obtain better clustering solutions.

Chapter 6: A novel supervised HAIS algorithm inspired by HIR is presented in chapter 6. The performance of HAIS is compared against well-established the Artificial Immune Recognition Systems (AIRS) algorithm. The main parameters and components of the supervised HAIS learning algorithm are evaluated to explain the behavior of the algorithm.

Chapter 7: A novel approach for integrating AIS concepts (using the HAIS supervised learning algorithm) with artificial neural networks (ANNs) is presented in this chapter. This chapter demonstrates that the concepts of AIS and ANN can be used closely in a novel way to achieve a robust, generalized and efficient supervised learning model.

Chapter 8: The conclusion is presented in this chapter, followed by the summary of the work conducted in this thesis. Finally, ideas for future work are presented.

1.4.3 Publications

The following five research papers have been written and published during the course of this research thesis [58-62].

1. W. Ahmad, A. Narayanan, "Humoral-mediated Clustering", *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2010),* 2010, pp. 1471-1481

2. W. Ahmad, A. Narayanan, "Outlier Detection using Humoral-mediated Clustering (HAIS)", *Proceedings of NaBIC2010 (IEEE World Congress on Nature and Biologically Inspired Computing),* 2010, pp. 45-52. Also appeared in the *International Journal of Computational Intelligence and Applications (IJCIA),* vol. 11, 2011

3. W. Ahmad, A. Narayanan, "Humoral Artificial Immune System (HAIS) for Supervised Learning", *Proceedings of NaBIC2010 (IEEE World Congress on Nature and Biologically Inspired Computing),* 2010, pp. 37-44. Also appeared in the *International Journal of Computational Intelligence and Applications (IJCIA),* vol. 11, 2011

4. W. Ahmad, A. Narayanan, "Principles and Methods of Artificial Immune System Vaccination of *Learning Systems*", in Pietro Liò, Giuseppe Nicosia and Thomas Stibor (Eds.), *Artificial Immune Systems, 10th International Conference, ICARIS 2011*, Cambridge, UK, 2011, pp. 268-281, Lecture Notes in Computer Science 6825, Springer 2011

5. W. Ahmad, A. Narayanan, "Population-Based Artificial Immune System Clustering Algorithm", in Pietro Liò, Giuseppe Nicosia and Thomas Stibor (Eds.), *Artificial Immune Systems, 10th International Conference, ICARIS 2011*, Cambridge, UK, 2011, pp. 348-360, Lecture Notes in Computer Science 6825, Springer 2011

Chapter 2

Artificial Immune Systems

The artificial immune system (AIS) paradigm has developed relatively recently in comparison to other nature-inspired computing techniques such as evolutionary/genetic algorithms, ant colony optimization (ACO) and particle swarm optimization (PSO). The main emphasis of this chapter is on the NIS and the inspirations it provides for developing AIS computational techniques in the field of machine learning. In order to form the basis for developing a novel AIS algorithm, it is necessary to provide a background of the NIS. Therefore, this chapter covers the principles, properties, functionality and core theories associated with NISs developed in recent years. Our understanding of the inner workings of the immune system is still not substantial, as we

are still at the exploration stage and it will take some time to establish a reasonably good understanding of the biological complexities involved. However, this chapter will provide a basic understanding of the NIS.

The next section (2.1) provides a description of the twined and complex behavior of the immune system. Section 2.2 maps some core NIS concepts to computational AIS. The main components of designing a framework of any nature-inspired algorithms, and especially an AIS, are discussed in section 2.3. Moreover, various immune system-inspired algorithms proposed in recent years are discussed in the context of our current understanding of an NIS. An extensive literature review of existing immune system-inspired algorithms is presented in section 2.4. The current state of AIS and various suggestions on the way forward constitute section 2.5, and finally a summary and conclusion of this chapter is provided in section 2.6.

2.1 Introduction to the Natural Immune System (NIS)

In 1882, the Russian zoologist Ilya Metchnikoff was strolling along the beach and collected a tiny transparent larva of starfish. He pierced the larva with a rose thorn and returned home. Next day, while walking on the same path, he noticed that minute cells were covering the thorn as if they were trying to engulf the thorn. In fact, those cells were attempting to defend the larva by ingesting the invader (thorn). This phenomenon later became known as ‘phagocytosis’. In 1908, for his work in this field, Metchnikoff was awarded a Nobel Prize in Medicine [63]. His rose thorn ‘experiment’ suggested that living organisms possess a mechanism that fights against outside invaders.

Throughout the life of humans, our body gets exposed to different types of diseases. In response to these diseases, humans have evolved a complex system called an immune system. Once the body becomes immune to some specific disease, it remains free from it almost for the entire life span. On the other hand, our immune system also protects us from self-altered cells which can develop from injuries or infections. The immune system recognizes those altered cells and destroys them: a good example is cancer cells growing inside body. Therefore our immune system not only protects us from the outside invaders but also from the inside invaders. An immune system works on two major levels one is an innate immune system and the second is acquired or adaptive immune system. The multi-layered structure of NISs can be seen in Figure 2.1 (taken

from [64]). Our skin provides us with the first layer of defense against invaders. Fluids such as saliva, sweat and tears contain detrimental enzymes, constituting the second layer, and are called physiological barriers. Innate and adaptive immune systems comprise the third and fourth layers (barriers) respectively.

Antigens are any particle (coats, capsules, cell walls, toxins) of bacteria, viruses and other micro-organisms. Simply, antigens are the substances which are detected by immune systems and subsequently trigger an immune response. This process is called immunogenicity. All immunogenic substances are antigens, but not all antigens are necessarily immunogenic. For example, haptens (e.g. penicillin) are antigens but are not immunogenic unless coupled to a larger carrier molecule [63]. Antigens are generally of two types: exogenous antigens and endogenous antigens. Exogenous antigens are microorganisms, pollens, drugs and pollutants that enter the host from the external environment. On the other hand, endogenous antigens are found within the host, and can trigger an immune response under various circumstances; cancer cells are an example of such antigens.

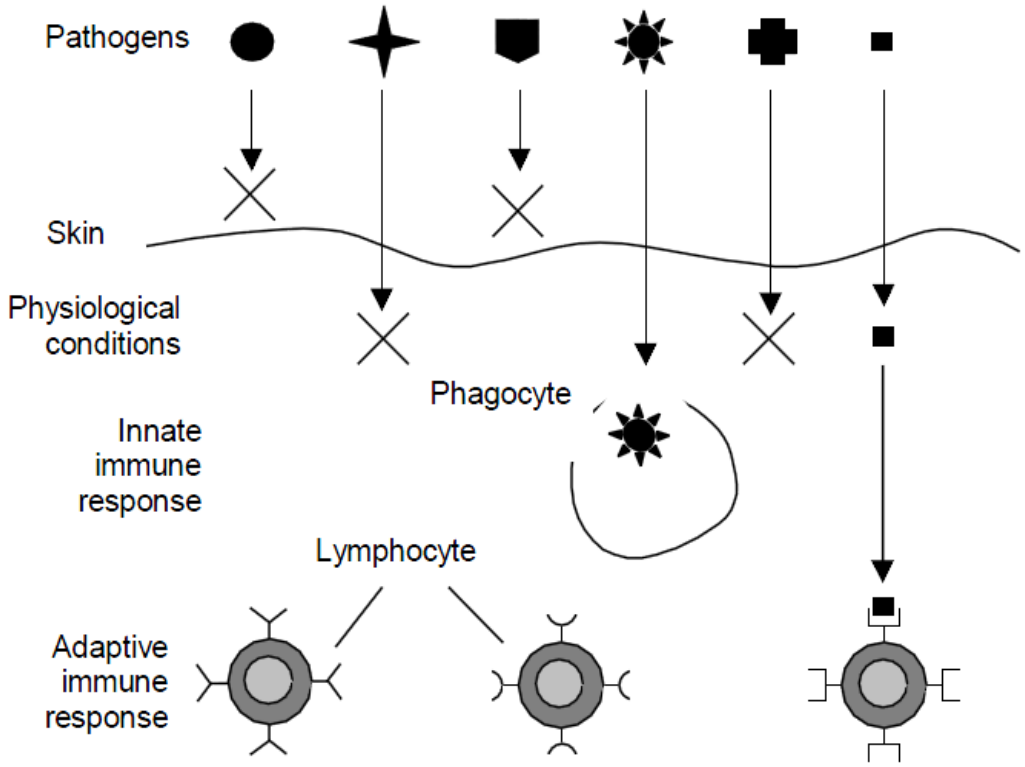


Figure 2.1: Multi-layered structure of the NIS [64]

2.1.1 Cells and Organs of the NIS

Human immune systems consist of various structurally and functionally different organs which are dispersed all over the body [63]. Based on their functionality, these organs are classified into two types: the primary lymphoid and secondary lymphoid organs. The main task of the primary lymphoid is to provide an appropriate environment for lymphocyte maturation, whereas the secondary lymphoid traps and presents antigens to mature lymphocytes. The thymus and bone marrow constitute the primary lymphoid organs, whereas lymph nodes, the spleen, and various mucosal associated lymphoid tissues (MALT) constitute the secondary lymphoid organs [63]. The thymus, as seen in Figure 2.2, is an example of a primary lymphoid organ and is located just above the heart. It is responsible for antigen-independent maturation and development of T-lymphocytes, which regulate cell-mediated immunity. In mammals, immature B-cells proliferate and differentiate in bone marrow through a process called B-cell maturation. Secondary lymphoid organs such as lymph nodes, the spleen, adenoids, tonsils and lymphoid patches of the guts and appendix are the areas where mature T or B-lymphocytes may interact with antigens and form the basis of antigen-dependent differentiation.

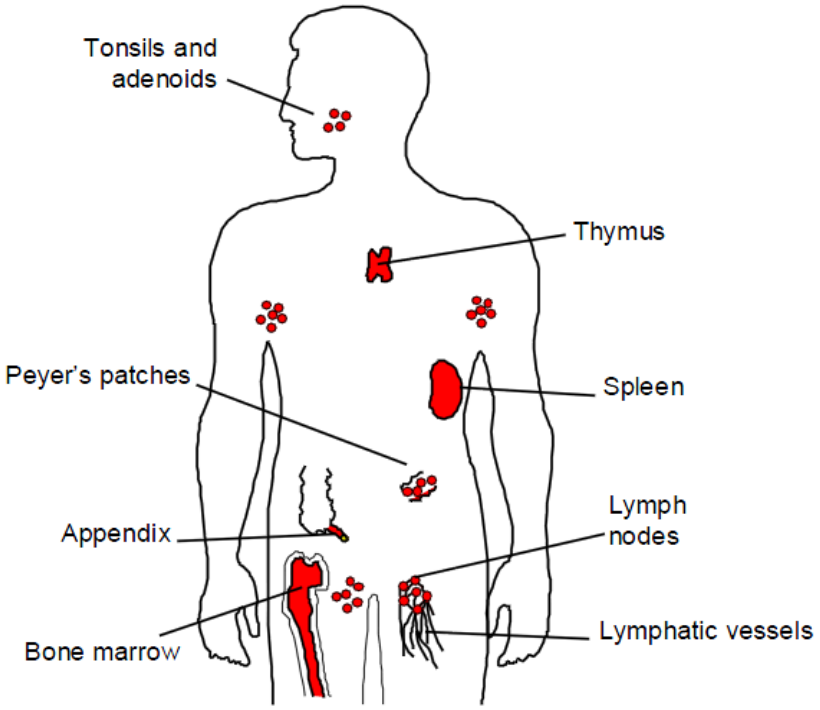


Figure 2.2: Organs of the human immune system [64]

2.1.2 Innate Immune System

The innate immune system is the one we have at birth, and it is likely that it stays the same throughout our lifetime. It is non-specific (i.e. not tuned for any particular infection) and is the first level of defense. One of the main components of the innate immune system is leucocytes, or white blood cells, which move throughout the body and capture invading micro-organisms and other foreign particles. Leucocytes are formed in the bone marrow from stem cells. Most leucocytes do not divide or reproduce. Typical leucocytes are natural killer cells and phagocytes (macrophages, neutrophils and dendritic cells). Dendritic cells (DCs) are important not only for their location (in contact with the external environment, such as in the skin and mucus) but also for their function, since they link the innate and adaptive immune systems. These DCs, also known as antigen presenting cells, help to regulate the adaptive immune system via the interaction of T-cells (which are also an important component of the adaptive immune system). If the innate immune system cannot deal with a pathogen, the adaptive system is triggered.

2.1.3 Adaptive Immune System

The main components of an adaptive immune system are white blood cells also known as lymphocytes. Lymphocytes normally stay in a passive state until they encounter specific molecules called antigens. Lymphocytes are normally one of two types: class T or class B. B-lymphocytes secrete proteins that bind to the antigens, helping the immune system to destroy antigens, whereas T-lymphocytes perform a variety of functions including recognition and killing of cells that are bearing non-self molecules on their surfaces, and also kill cancerous cells. Each lymphocyte can recognize only one type of antigen. Each antigen leaves genetic blueprints on B- or T-lymphocytes so that if similar antigens attack the body in the future they are recognized quickly by the lymphocytes. This phenomenon is called immunology memory. Immunity mediated by T-lymphocytes is known as cell-mediated immunity. Humoral immunity, on the other hand, is mediated by secreted antibodies produced in the cells of B-lymphocytes.

2.1.3.1 Cell-Mediated Immunity

Cellular immunity or cell-mediated response is carried out by T-cells, which do not produce antibodies. T-cells are produced in the bone marrow but they mature in the thymus. During maturation, T-cells express unique antigen-binding receptors. These

receptors cannot activate independently on interaction with an antigen – the antigen needs to be bound to a specific molecule, known as a class I or II MHC (major histo-compatibility complex). When a T-cell encounters an antigen with an MHC molecule it undergoes differentiation and the production of a memory cell occurs. T-cells can also be categorized into T-helper and T-cytotoxic cells. A T-helper cell gets activated when an antigen is associated with MHC II class, and when antigen is associated with MHC I class T-cytotoxic cells are activated. The typical functions of T-cells are killing tumor cells and the rejection of foreign tissue graft.

Some viruses and bacteria enter the host's cells and cause infection. Antibodies cannot enter these cells. The functionality of an MHC is to interact with these infectious cells and bind and display small proteins or fragments from the infecting viruses and bacteria. Later, these molecules are presented to T-cell to trigger an immune response. Therefore, one of the functions of an MHC is to signal that a cell is infected [63]. If the interaction between T-cell receptors and an antigen-MHC is of low affinity, T-cells will evolve co-receptors and other membrane molecules to enhance the stimulation level. Once T-cells have undergone the process of proliferation and differentiation, they (T-cells) produce T-memory cells and T-effector cells, just like a B-cell, but different in functionality. While the role of T-cells is also important in immune systems, T-cells are not discussed further in this thesis since its aim is to explore the inspiration of B-cell mediated immunity in machine learning.

2.1.3.2 Humoral-Mediated Immunity

Humoral components consist of B-lymphocytes or B-cells. These B-cells produce antibodies, which attach/bind to the antigens and destroy them. B-cells are produced in bone marrow and reside there until maturation. When they are released, they have a unique receptor on their membrane called an immunoglobulin (Ig) receptor, due to its link to a specific Ig super-family. When a B-cell encounters an antigen, it reacts in two ways: one is the production of a memory B-cell and the other is the production of an effector B-cell. The memory B-cell also has the Ig receptor on its membrane and keeps looking for more similar antigens. The plasma cell or effector B-cell loses the Ig receptor and grows larger in size and starts producing antibodies. These antibodies are released into free circulation in order to encounter the antigens (see Figure 2.3). The new B-cells that have not encountered any antigens are called naive cells. They have a

life span of about a week unless they interact with an antigen. These naive cells circulate in the blood and lymphatic system. At this stage, if a B-cell interacts with any antigens, this will again lead to the process of proliferation and differentiation, leading to the secretion of plasma cells (effector cells) and memory B-cells. The life span of memory B-cells is much longer than regular B-cells and they can live on for many years. This is one way of keeping track of the diseases (or viruses) we have encountered in the past; if they come back, the immune system will be well prepared to act on those, due to the presence of the memory cells. If the same pathogens attack the body, due to the presence of memory cells the immune response is more direct and faster than the first time.

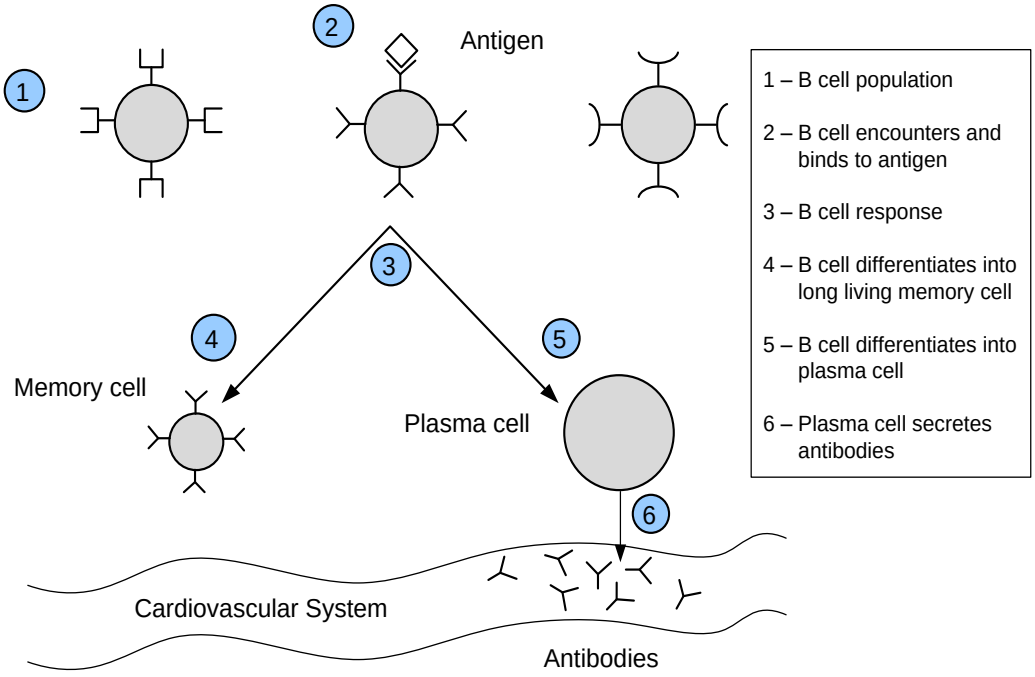


Figure 2.3: A snapshot of humoral-mediated response

Differentiation of B-cells (cell differentiation) can be of two types: the first is known as antigen independent differentiation and second is called antigen dependent differentiation. The B-cells that are matured in the bone marrow and then get into the blood and lymph are driven by the antigen independent differentiation process. When a B-cell encounters an antigen and produces plasma cells and memory cells, this is known as antigen dependent differentiation process. The antigen dependent differentiation process occurs in two stages: (1) cell activation and (2) proliferation and differentiation. In the case of antigen dependent differentiation, cell activation also occurs in two

different ways: (1) B-cell activation by T-cell independent antigens – this occurs on the surface level as the epitopes of antigens interact with the epitopes of the antibodies – and (2) B-cell activation by T-dependent antigens. This occurs when antigen-presenting cells interact with T-cells and those T-cells interact with B-cells. The first approach is more likely to be a physical interaction which leads to an immune response; the second is a chemical interaction between antigens and antibodies, leading to an antibody response.

When the antigen and antibodies interact with each other, they react to form a non-covalent bond, meaning that they are not involved in the sharing of the pairs of electrons. This bonding is considered to be weak and therefore a strong affinity interaction must exist in order to make the stable complex. These antibodies have multiple binding sites and each antibody can interact with one or more similar antigens. As more than one antigen can be attached to one antibody, high avidity compensates for low affinity. The interaction between antigen and antibody is very specific in nature, but sometimes certain antigens show cross-reactivity with unrelated antibodies and some antibodies show cross-reactivity to unrelated antigens, but this may only occur when different antigens share the same epitope or unrelated epitopes have similar chemical properties. This cross-reactivity can lead from tissue damage to adverse autoimmune reaction. As stated earlier, each antibody has more than one antigen-binding site. As the number of antigens increases, antigens with less affinity are displaced and replaced with antigens with higher affinity.

Antibodies are produced during the process of proliferation (in plasma cells) and are subsequently released into the bloodstream. These antibodies are also called Igs, which are classified into five major categories: IgG, IgM, IgA, IgD, and IgE. In the experiments of Porter and Edelman in the 1950s, it was revealed that IgG was composed of two identical heavy chains (H) and two identical light chains (L). IgG also has two regions: the variable region and the constant region. Variable regions (or V regions) are responsible for antigen recognition, whereas constant regions (or C regions) are responsible for a variety of effector functions such as complement fixation [64].

Jerne [52] proposed ‘immune network theory’, where antibodies trigger an immune response not only when they interact with antigens but also with other antibodies. Immune network theory states that the V region in antibodies responsible for antigen

interaction can sometimes bind with other V regions to form an interconnected network called the idiotype network. According to this model, each arm of antibody molecule possesses one paratope (an antibody-combining site) and a small set of idiotopes. Lymphocytes may respond positively or negatively to a recognition signal. A positive response leads to cell activation and differentiation whereas a negative response leads to tolerance or suppression. This process of suppression and activation can be seen in Figure 2.4.

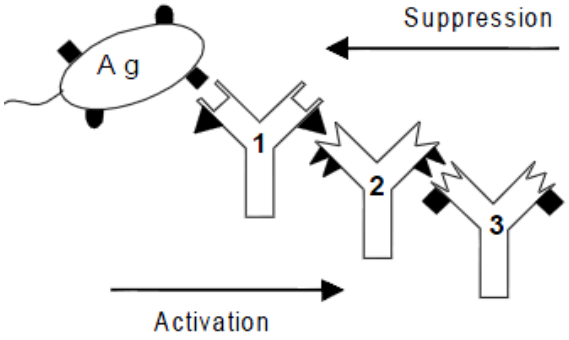


Figure 2.4: Suppression and activation phenomena in antibodies [64]

The immune system works on the phenomenon of negative selection. This is one of the ways of differentiating among ‘self’ and ‘non-self’. The immune system destroys all the antibodies which are similar to self to avoid a self-destructive immune response and keeps anything that is alien or non-self. This record of non-self keeps updating as new antigens or viruses are introduced to the immune system. This negative selection process takes place in the thymus. The B-cells that bind to self are destroyed and only those that do not bind to any of the self-cells are allowed to leave the thymus and circulate all over the body to detect and kill the antigens.

Only those antibodies that interact with antigens are selected for proliferation and differentiation and all others remain in an inactive or passive state. The main feature of clonal theory is that all the proliferated cells are identical copies of the original antibody with a mutation rate which is dependent on the affinity measure (similarity measure) between the antigen and the antibody. Again, all the cells that interact with self are destroyed.

2.1.4 Vaccination

Edward Jenner and Louis Pasteur were the first to develop vaccination for human diseases [63]. Vaccination is considered to be an efficient and low-cost disease prevention technique. One of the shining examples of vaccination is the eradication of smallpox disease worldwide; there has not been a single case of it reported since 1977. Recent advancements in immunology, a better understanding of T-cell and B-cell receptors, and their interaction with antigenic determinants and genetic engineering have all contributed in improving vaccines to generate maximum immune response.

Vaccination is a process where inactive or weakened forms of pathogens are introduced into healthy human beings to stimulate a defensive response without leading to the disease [65]. A vaccine typically activates an immune response in the form of the generation of antibodies, which are cloned and hyper-mutated to bind to antigens (fragments of pathogens). Vaccines lead to the production of antibodies so that the NIS is primed if in the future a stronger version of the pathogen is encountered. Vaccines are categorized by composition and formulation (how they are derived, how they are used, and how their effects are mediated). For example, the tetanus vaccine is produced from the toxic chemicals extracted from the tetanus pathogen, as are the vaccines for hepatitis B and diphtheria. There are different ways to obtain vaccines [65], such as:

1. **Live attenuated vaccines:** consist of a weakened form of the infectious agent itself. The measles, mumps, rubella and some polio vaccines are examples of live attenuated vaccines.
2. **Inactivated vaccines:** involve inactivating infectious agents by heat or by chemicals and later injecting them into healthy people. The influenza (flu), rabies, hepatitis A and pertussis are examples of inactivated vaccines.
3. **Acellular and subunit vaccines:** typically consist of proteins extracted from infectious agents. The hepatitis B and plague immunization are examples of acellular and subunit vaccines.
4. **Toxoid vaccines:** some pathogens, when invaded, produce toxins that get into the bloodstream. The tetanus and diphtheria vaccines are examples of toxoid vaccines.

2.1.5 Danger Theory

As stated earlier, immunologists originally believed that the immune system works on the principle of distinguishing between self and non-self. However, this model started to encounter problems when immunologists recognized that the activation of T-cells also depends on signals from other cells called antigen-presenting cells (APCs). In 1994 Matzinger [66] proposed her 'danger theory'. This theory states that APCs such as DCs are automatically activated through an alarm called a danger signal. These activated APCs then provide a co-stimulatory signal to T-cells, which subsequently control the activation of adaptive immune system. These danger signals are emitted by cells that are damaged due to the attack of antigens. While collecting antigens in the peripheral parts of the body, DCs detect these signals and make decisions regarding generating safe or danger signals. If DCs detect a dangerous environment, they present collected antigens as well as danger signals to initiate an immune response. Danger theory has serious consequences for the way we perceive the inner workings of the immune system. The generation of a danger signal by the innate immune system as well, controlling the activation of T-cells, eliminates the discrimination between the adaptive and innate immune systems. Another important impact of this theory is the replacement of the notion of self and non-self with the danger and not-danger metaphor.

2.1.6 Summary

There are a number of features of the NIS which capture the attention of AIS and especially machine learning researchers. Some of these characteristics have been highlighted by Hunt and Cook [11]:

Generation of Antibodies: One of the most outstanding aspects of the NIS is its ability to generate millions of antibodies from a few antibody genes. This feature helps humans survive various deadly viruses and pathogens which have never been encountered before.

Diversity: The NIS encourages diversity. The functionality of the NIS through antibodies and other processes is not to find global optima; instead, it helps evolve antibodies, which can bind to a variety of antigens. At the time of proliferation and differentiation, the main objective of the NIS is to achieve local best solutions only.

Distributed System: The NIS is a distributed system with no central control. The cells and organs of the NIS are distributed throughout human body.

Dynamic System: The NIS is an effective dynamic system which helps mount an appropriate response to ever-changing and challenging pathogens. Once an antigen enters into the system, the NIS with the help of T helper cells triggers an immune response that results in the generation of millions of antibodies to adapt to an invading pathogen. The NIS also is equally efficient in identifying and destroying self-altered cells.

Self-Organizing Memory: The NIS possesses a self-organizing memory that is maintained through the generation of memory cells. It is one of the stand-out features of the NIS and the focus of many machine learning researchers. It is this self-organizing memory that helps the NIS recognize and mount an appropriate immune response in the event of the same pathogens invading the body again (secondary immune response).

Noise Tolerance: The NIS is tolerant to noise. The immune system response is only triggered if the interaction between antigen and antibodies exceeds a certain threshold. In addition, the NIS has the capability to re-classify itself in light of new information regarding pathogens.

2.2 Basic Concepts of an Artificial Immune System (AIS)

We have explained in section 2.1 some of the main processes involved in NISs. In order to map processes and information available in an NIS to an AIS, a few considerations are necessary, regardless of the area of application:

1. How to represent antigens, antibodies and B-cells?
2. What are memory cells in computational AISs?
3. How to calculate affinity between antigen and antibody receptors?
4. How to perform clonal selection?
5. What does hypermutation mean?

In this section we answer these questions in the context of AISs in a machine learning paradigm. All the unseen data instances are usually considered as antigens. In the example of a cancer dataset, the task is to classify all data instances/records into either a normal group or cancer group, with or without seeing an original class label. Regardless

of biological differences, there is little to distinguish in the use between antibodies and B-cells in the literature. Antibodies are data items which we have already seen and classified, whereas B-cells can be regarded as data clusters. Memory cells are also data items which we have already seen. The main difference between antibodies and memory cells is that the former are generalized forms of already seen data and the latter are specializations of seen data instances. More discussion regarding AIS antibodies and memory cells will be provided in the coming chapters. The antigens and antibodies must be represented in the same way, e.g. as a number of continuous or discrete variables or features.

The affinity or similarity measure is one of the most important factors in designing an AIS and it depends on the representation scheme of antigens and antibodies. For example, consider an antigen = [1 1 0 0 1] and an antibody = [0 1 0 1 1]. There is a similarity score of 3 as there are three matching indices. If the representations of data items are real variables, various other distance measures can also be applied such as Euclidean distance, Manhattan distance, Mahalanobis distance and Hamming distance. Essentially, these measures return a magnitude of similarity or dissimilarity, which can be considered as the affinity between an antigen and an antibody.

Mutation in an AIS is the same as that performed in genetic algorithms. For example, for binary string, one or more bits are randomly flipped, and in real-value string, one or more values are changed to other values. There are different ways that mutation can be implemented in an AIS. For example, the mutation rate can be higher or lower depending on the affinity between antigen and antibodies. When a new data instance is introduced into an AIS, it is compared against all existing antibodies, and those antibodies with a threshold higher than some specified level (affinity) get stimulated and activated. Once an antibody is activated it undergoes clonal selection. Clonal selection is the process of generating more antibodies by activated antibodies to achieve a higher affinity level between antigen (new data instance) and antibodies. The process of fine-tuning antibodies receptors (moving of antibodies towards new data instance) towards an antigen is called affinity maturation.

2.3 AIS Framework

Put simply, a framework is a specific set of instructions which help design any computational algorithm. De Castro and Timmis [67] proposed a framework for AIS. In the case of other nature-inspired computational techniques (e.g. neural networks and genetic algorithms) frameworks already exist which can help considerably in the development of AIS systems. For instance, one framework for designing an artificial neural network [13, 18] consists of artificial neurons, interconnecting weights on neurons, and a learning algorithm [18, 67]. An artificial neural network can be designed by arranging together artificial neurons in various layers (input layer, hidden layer and output layer). During the training or learning process, the artificial neurons undergo an adaptive process by which weights associated with those artificial neurons are adjusted and quantified and finally knowledge is acquired from the system. In the case of genetic/evolutionary algorithms, an initial population of chromosomes is generated, which during the learning process undergoes reproduction, genetic variations and selection. As a result of this learning process a population of evolved chromosomes (individuals) arises. In summary, the framework for designing an evolutionary algorithm is the generation of an initial population of chromosomes, and the procedures for reproduction, genetic variations and selection. The authors [67] derived some simple rules for designing any nature-inspired algorithm. It must have:

- "A representation of the components of the system
- A set of mechanisms to evaluate the interaction of individuals with the environment and each other. The environment is usually simulated by a set of input stimuli, one or more fitness function(s), or other mean(s) and
- Procedures of adaptation that govern the dynamics of the system, i.e. how its behavior varies over time" [67].

The framework for designing AIS algorithms is represented as a layered approach in Figure 2.5. The AIS system consists of three components: representation, affinity measure, and immune algorithms [67]. To build an AIS, the first step is to know the application domain. The application knowledge will help the representation of the AIS system. The affinity measure also depends on the representation of AIS system. One or a combination of distance measures can be used for the affinity measure. Each of these

distance measures have their own biases [54, 55] and an affinity measure must therefore be selected with great care as it can affect the final solution obtained by the system. Finally, the final layer is the selection of appropriate immune system-based algorithms. Some of the proposed AIS algorithms are negative selection, positive selection, clonal selection, and the immune network algorithm [10]. The details regarding these algorithms are presented the next section.

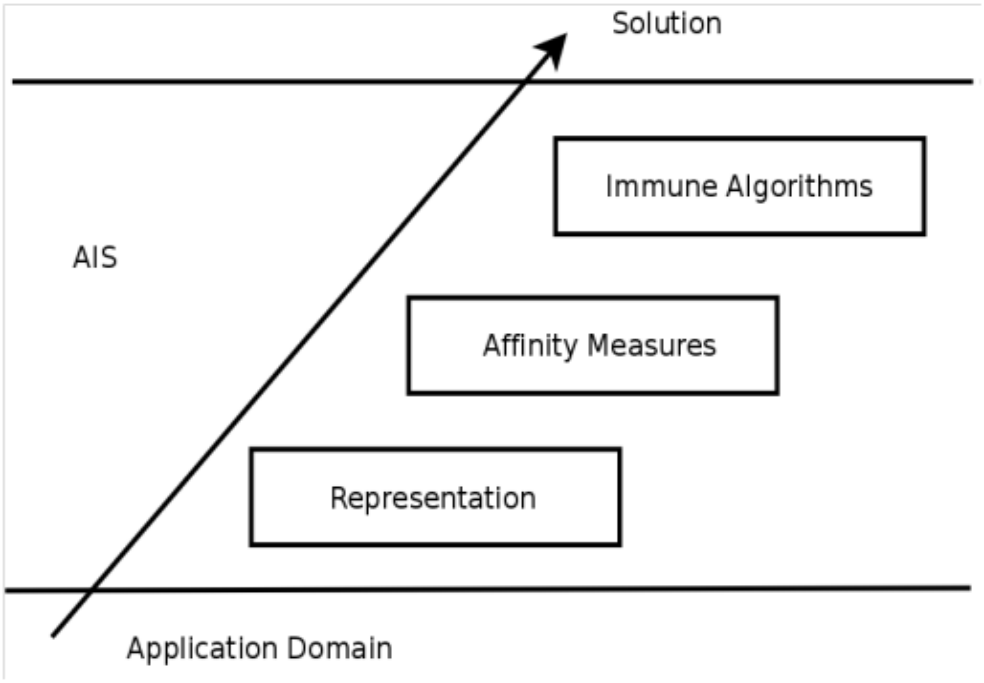


Figure 2.5: AIS layered framework [10]

The ‘Representation’ and ‘Affinity Measures’ steps in the above framework can be considered as being part of the ‘Immune Algorithms’ step. According to Jain [54], each algorithm is designed to solve a specific task and therefore these algorithms implicitly or explicitly impose structural constraints on the data. In addition, each algorithm has parameters which needed to be optimized to obtain desired solutions. Therefore, the above framework presented in Figure 2.5 can be revised into that presented in Figure 2.6.

The NIS is a complex and interconnected system comprising of various micro-level processes. Some of the main features currently believed to be important in the NIS are: negative selection, clonal expansion, affinity maturation, immune network theory, and danger theory. AIS researchers have drawn inspiration from these processes and developed various algorithms inspired from one or a combination of them. An in-depth

examination of the existing literature reveals that AIS algorithms can generally be classified into five distinct groups. These algorithms, their inspiration, and other related details are discussed in the following section.

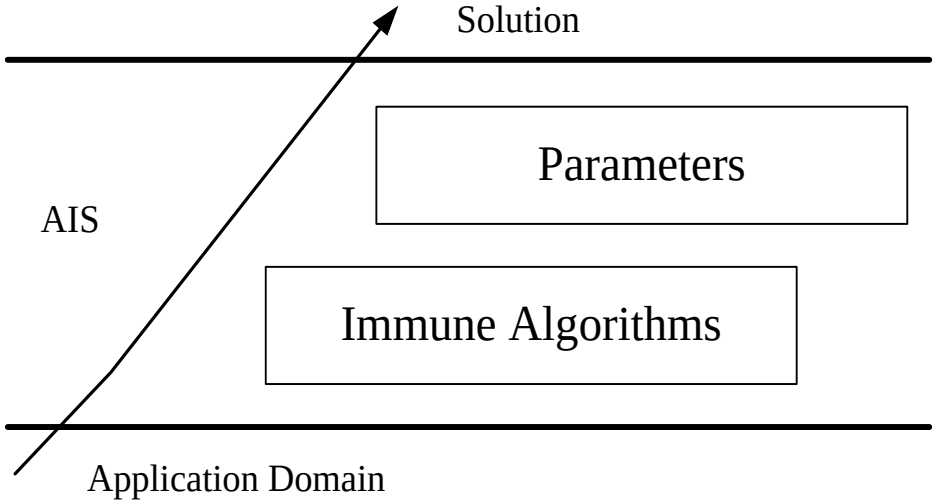


Figure 2.6: Revised AIS framework

2.3.1 Negative Selection Algorithm

One of the earliest AIS algorithms was proposed by Forrest *et al.* [68] and was inspired by the negative selection occurring in the thymus on T-cells during the T-cell maturation process. This algorithm is tested on anomaly detection data and can be divided into two separate phases: training and testing. The generation of valid detectors using self data constitutes the training phase, whereas the evaluation of those generated detectors on unseen data instances constitutes the testing phase. In the first phase, the algorithm generates detector strings, which are a complement of self strings. The self strings can be seen as the normal state of the system, and generated detectors would capture/detect only the deviations from the self state. The first phase ideally finishes when all the generated detectors have not been able to detect self strings. In the second phase, new and unseen self and non-self strings are introduced into an already learned system, comparing them against already generated detectors to assess the quality of generated detectors. The basic components of negative selection algorithm can be seen in Figure 2.7, where (a) and (b) represent the training and testing phases respectively.

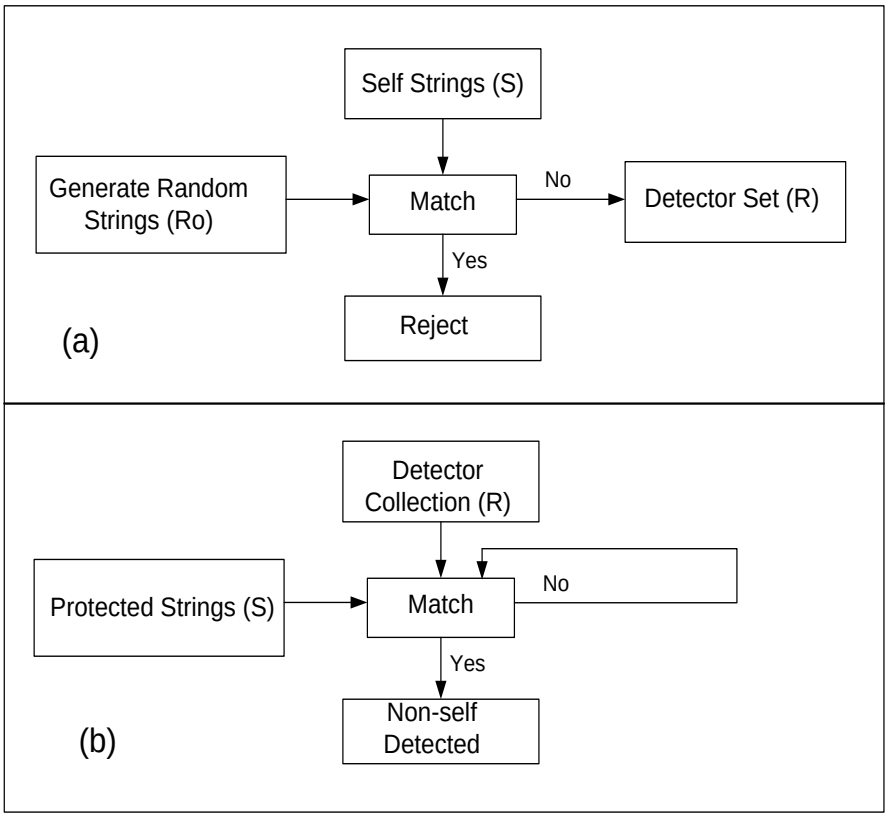


Figure 2.7: The two phases of the negative selection algorithm proposed by Forrest [68]

In recent years, a series of theoretical studies conducted by Stibor *et al.* [69-73] on the negative selection algorithm have highlighted a number of drawbacks, which include its poor performance in comparison to well-established statistical models, as well as the difficulties and challenges the negative selection algorithm faces in generating non-self detectors in high-dimensional space.

2.3.2 Clonal Selection Algorithm

CLONALG [74] is an example of a clonal selection algorithm which was originally specified for binary character recognition and engineering optimization but which can be adapted for unsupervised learning. This algorithm is inspired by the B-cell activation process, which results in the generation of plasma cells and antibodies that are subsequently released into the bloodstream to capture similar antigens. The main components of this algorithm are antibodies, cloning and hypermutation, and affinity measure and selection. An overview of this algorithm can be seen in Figure 2.8. CLONALG can be described as follows, where antibodies are regarded as cells [75]:

Initialize: *Prepare an initial random antibody pool.*

Loop: *Present the antigens to the antibodies and calculate affinity values for each antibody depending on the similarity between the antibody and the antigens.*

Select: *A set of antibodies are selected from the entire antibody pool that have the highest affinity with the antigen.*

Clone: *This set of selected antibodies is cloned in proportion to their affinity (the higher the affinity, the more an antibody is cloned).*

Mutate (affinity maturation): *Mutate the clones in direct inverse proportion to their affinity (the lower the fitness, the more a cloned antibody is mutated).*

Measure clonal affinity: *Calculate affinity values for each cloned antibody depending on the similarity between the cloned antibody and the antigens.*

Select candidates: *The antibodies (original and cloned) with the highest affinity are selected for survival in the next generation.*

Diversify: *Add a number of newly generated random antibodies.*

Return: *to Loop until some termination condition is satisfied.*

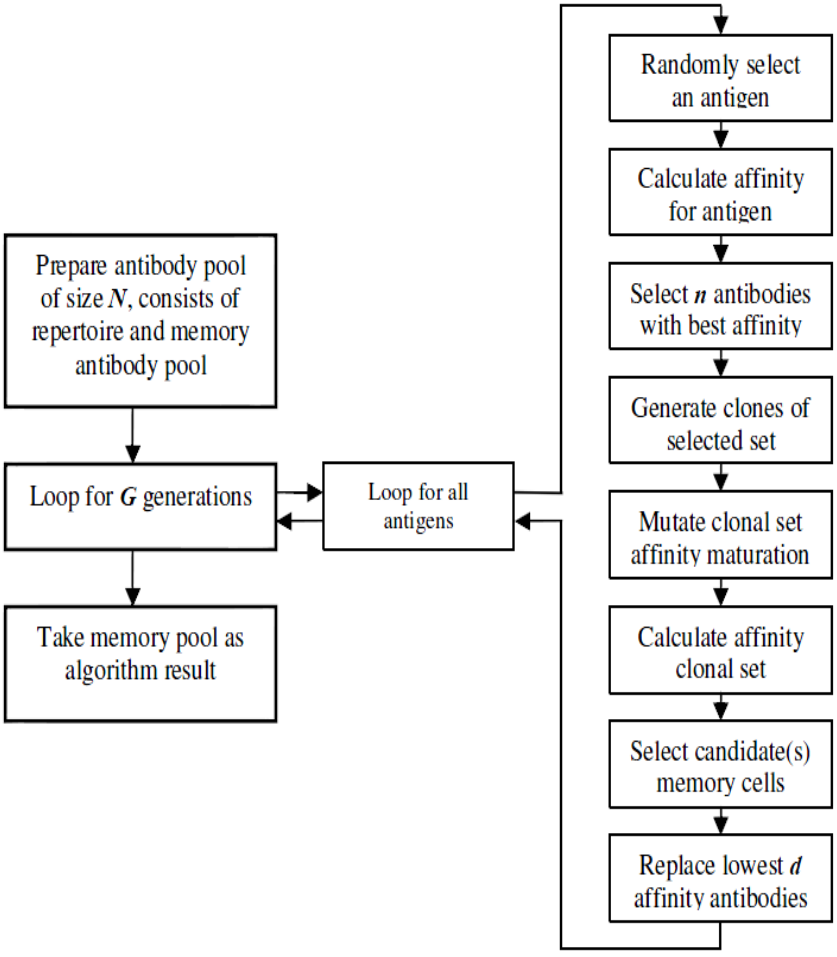


Figure 2.8: Overview of CLONALG [75]

This generic algorithm can be fine-tuned depending on the problem being dealt with. For instance, if binary classification is sought, the class of antigen can be used in the affinity measure to produce just two antibodies that are highly tuned to their respective antigens. The algorithm reflects several NIS principles, including the match between antigens and antibodies and the somatic hypermutation and clonal selection of those antibodies that match antigens. CLONALG shares many similar features with evolutionary algorithms such as mutation and the selection mechanisms performed in evolving a population of antibodies. Therefore, theoretical approaches applicable to evolutionary algorithms can also be applied to clonal selection algorithms [76]. Clark *et al.* [77] proposed the Markov Chain Model of the clonal selection algorithm called B-cell algorithm (BCA) for function optimization, proving its convergence.

2.3.3 Immune Network Algorithm

In his landmark paper, Jerne [78] proposed that antibodies trigger an immune response not only when they interact with antigens but also with other antibodies. This immune response is known as the idiotype network. Antibodies may respond either positively or negatively to a recognition signal. A positive response leads to cell activation and differentiation whereas a negative response leads to tolerance or suppression. De Castro and Zeben [79] proposed an AIS algorithm based on immune network theory called aiNet. The main difference between this algorithm and other AIS algorithms is the idea that the components of an AIS not only interact with antigens but also with each other to form stable network, in the form of memory cells. The main characteristic of this algorithm is that it can be used for data reduction as well as cluster analysis. Data reduction is achieved through generation of memory cells and clustering is performed on the obtained memory cells using a minimum spanning tree (MST). This is a two-step algorithm where immune system concepts are used to obtain a set of memory cells and those memory cells are later used for cluster analysis. This algorithm is a modified version of the clonal selection algorithm discussed earlier. RAIN is another example of immune network algorithms and was proposed by Timmis and Neal [80]. Figure 2.9 presents a flowchart showing the main steps of the immune network algorithm.

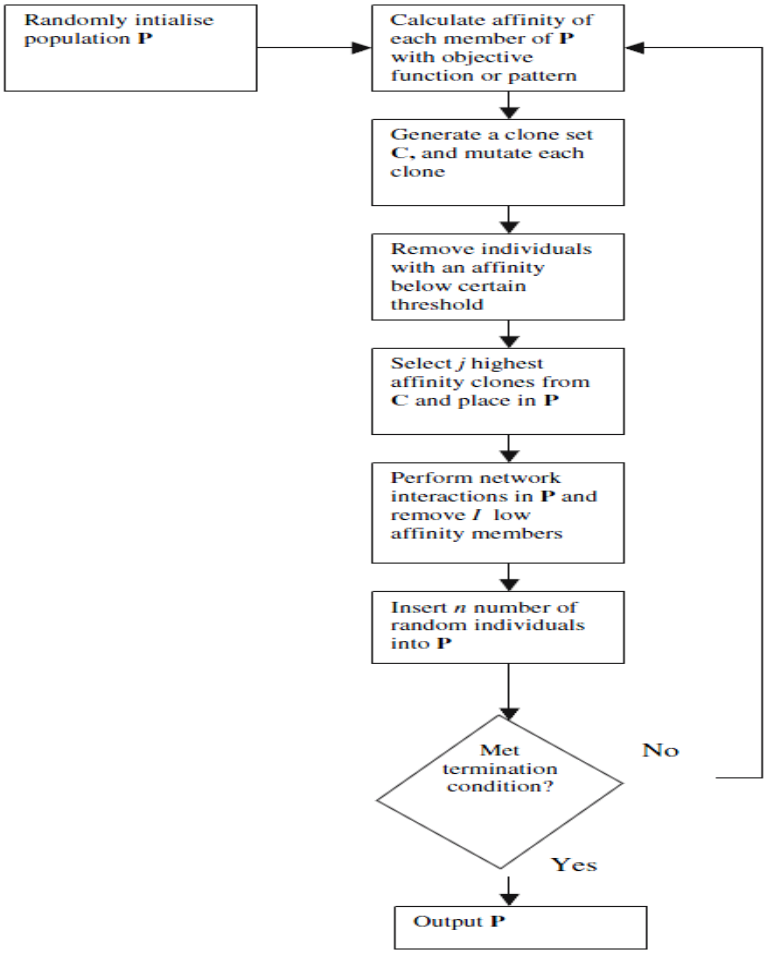


Figure 2.9: Flowchart of the immune network algorithm [33]

One of the major drawbacks of both aiNet and CLONALG for unsupervised learning is their inability to discriminate self patterns from non-self patterns at the learning stage. For example, aiNet only uses the first phase (learning phase) to generate memory cells, which are regarded as an abstraction of input data patterns and only to be seen as data reduction process. It is in the second phase where cluster labels (in terms of binary classification: self and non-self labels) are attached to those memory cells by performing a minimum spanning tree (MST) process. Put simply, these approaches only recognize the structural features of the data, and are unable to recognize and extract data partitioning.

2.3.4 Danger Theory Algorithm

Matzinger's [66] danger theory has received much attention from researchers. As noted above, it suggests that the immune system does not differentiate self from non-self, but what is harmful and what is not harmful to the body. According to this theory APCs

have the capability to activate an alarm signal called a 'danger signal' when harmful invaders enter the body. These APCs stimulate T-cells, which subsequently control an adaptive immune response. Secker *et al.* [81] proposed an algorithm to explore the relevance of danger theory to web mining by investigating the use of danger signals. This has been extended by Greensmith and her team with the dentritic cell algorithm (DCA) [82-86], using the concepts of the danger theory: danger signals, safe signals, and pathogen-associated molecular pattern (PAMP) signals. The importance of this work is that it removes the need to define what self or non-self is, although it is necessary to define what the danger, safe, and PAMP signals are. The basic DCA is shown below [87]:

input: S = Data to be labelled safe or dangerous
output: D = Data labelled as safe or dangerous
begin
Create an initial population of dendritic cells (*DCs*), ***D****, and a set* ***M*** *to contain* migrated *DCs*
Forall data instances in *S*, do:
Create a pool of **P,** randomly selected from D
forall DCs in **P**, do:
Add data item to DCs collected list
Update danger, PAMP and safe signal concentrations
Update concentrations of output cytokines
Migrate the DC from **D** to **M** and create a new DC in **D** if: concentration of co-stimulatory molecules is above a threshold
end
end
forall DCs in **M**, do:
Set DC to be semi-mature if output concentration of semi-mature cytokines is greater than mature cytokines otherwise, set as mature
end
forall data instance in **S**, do:
Count number of times data instance is presented by both mature DC and a semi-mature DC.
Label data instance **safe** if presented by more than semi-mature DCs than mature DC's, otherwise label as **dangerous**
Add data instance to labelled set **M**
end
end

2.3.5 Artificial Immune Recognition System (AIRS)

The leading work in AIS supervised learning algorithms is associated with the Artificial Immune Recognition System (AIRS) of Watkins and Timmis [88], which uses the concepts of artificial recognition balls (ARBs), resource limitation, memory cells, and hypermutation. ARBs are essentially B-cells supplemented with information on the

resources available in the system. The idea of ARBs in AIS was initially proposed by Timmis *et al.* [89] and further fully utilized in [80]. AIRS adopts a one-shot approach in that learning patterns (antigens) are allocated to the closest matching ARB in the pool of ARBs, followed by a competitive stage in which the ARBs either survive or die depending on their fitness with regard to capturing antigens of the right class. Resources are re-allocated throughout the ARBs depending on which ARBs survive or die. Memory cells are produced from the surviving ARBs. At the end of the one-shot approach, the memory cells adopt a k-nearest neighbor (KNN) voting method by presenting test samples to all memory cells and reporting the stimulation values returned by each memory cell. The AIRS algorithm is presented below [88]:

1. **Initialization:** Create randomly memory pool (**M**) and the ARB pool (**P**).
2. **Antigenic Presentation:** for each antigen do:
    a) **Clonal Expansion:** For each element of **M** calculate their affinity to antigen from the same class as antigen. Select highest affinity memory cell (mc) and clone mc in proportion to its affinity and add it to the set of ARBs (**P**)
    b) **Affinity Maturation:** Perform mutation on each of mc in ARB of this highest affinity. Place each mutated ARB into **P**.
    c) **Metadynamics of ARBs:** Process each ARB through the resource allocation mechanism. Calculate the average stimulation for each ARB, and check for termination condition.
    d) **Clonal Expansion and Affinity Maturation:** Clone and mutate a randomly selected subset of the ARBs left in **P** based in proportion to their stimulation level.
    e) **Cycle:** While the average stimulation value of each ARB class group is less than a given stimulation threshold repeat from step 2.c.
    f) **Metadynamics of Memory Cells:** Select the highest affinity ARB of the same class (as antigen). If the affinity of this ARB is greater than the previously selected best memory cell mc, then add the candidate (mc-candidate) to memory set **M**. And, if the affinity of mc and mc-candidate is below the affinity threshold, then remove mc from **M**.
3. **Cycle:** Repeat step 2 until there are no antigens left.

AIRS adopts several immune system concepts and the results reported are competitive with other traditional supervised learning algorithms. The AIRS uses some of the core concepts of immune system such as clonal selection, affinity maturation, hypermutation, antibodies and memory cells. It is generally regarded and classified as a clonal selection algorithm, as both the AIRS and clonal selection algorithms share inspirations from

common principles of the NIS. But it is important to describe the AIRS separately as it is a specialized supervised learning algorithm that is inspired by the NIS.

McEwan and Hart [90], evaluated some of the parameters used in AIS algorithms (clonal selection, affinity maturation, memory cells) and discussed their defects using the AIRS algorithm. In their experimental work, clonal selection is replaced with a deterministic search operator. The authors' experimental results proved that deterministic search operators can perform as well as clonal selection and affinity maturation. They ran experiments on the data reduction capabilities of AIRS and concluded that a simple k-means algorithm appeared to have better data reduction capabilities than AIRS. Moreover, another flaw of the AIRS algorithm mentioned in their work is its inability to generate memory cells from data without following the density structure present in the original data.

2.4 AIS Algorithm Groupings

The early researchers of AIS considered that an immune system-inspired algorithm can only work in the pattern recognition domain due to its alignment with the human immune system and its pathogen recognition and elimination capabilities. The researchers have been using AIS algorithms on a number of different application domains such as computer security [86, 91], clustering/classification [67, 74, 88, 89, 92], optimization [74, 93] and robotics [94-97]. Hart and Timmis [10] presented a general review of application areas where AIS has been applied. The authors, based on a natural grouping of published work, classified AIS application areas into 12 distinct groups (Table 2.1) [10]. A reflection of these groupings can also be found in [98].

Table 2.1: Main application areas of AIS

Major	Minor
Clustering/Classification	Bio-informatics
Anomaly Detection	Image Processing
Computer Security	Control
Numeric Function Optimization	Robotics
Combinatorial Optimization	Virus Detection
Learning	Web Mining

With careful observation it can be seen that some of these groups can be merged to make a super-group. As discussed by the authors [10], some of the groups such as virus detection and computer security can be considered as a part of anomaly detection. Numeric function optimization and combinatorial optimization can both be regarded as 'Optimization'. Image processing and web mining can be regarded as clustering/classification, which can further be grouped into the super-group of 'Learning'. Therefore, based on this discussion, these 12 groupings can be re-classified into 3 main groups, namely Anomaly Detection, Optimization and Learning [10].

2.4.1 Anomaly Detection

Anomaly detection has been a favorite domain for AIS researchers. Normally, only a single class is available to train the network. The main objectives of such immune-inspired clustering algorithms are to learn from the example of one class (usually what is considered to be the normal behavior of the system) and generate a set of detectors that can identify any deviation in the system (deviation from the normal functionality of the system). The early work of Forrest *et al.* [68] provides the basis for using AIS algorithms for intrusion detection. This work was further extended by Kim and Bentley in [99], who incorporated a clonal selection algorithm with a negative selection algorithm to reduce false positive rates. Dal *et al.* [100] proposed an AIS intrusion detection algorithm based on negative clonal selection (generating non-self detectors based on self detectors). This is an online learning algorithm, where memory cells are used at the testing phase, when unseen data is presented to the trained model. The closest stimulating detectors undergo cloning and mutation to select best affine detector as a memory cell. Yue *et al.* [101] proposed an AIS algorithm for filtering spam e-mails. In this algorithm, a modified version of aiNet [79] is used to handle an incremental clustering problem as well as mining spam data streams. In the same domain, Secker *et al.* [102] proposed an AIS-based algorithm (AISEC) for the online classification of e-mails into two categories: interesting and non-interesting. The proposed algorithm has performed as well as naive Bayesian systems. An extension and further evaluation of AISEC (e-mail classification system using AIS) is presented in [103].

A multi-level AIS algorithm for detecting anomalies in data is presented by Dasgupta *et al.* [104], which integrates different immune system concepts such as negative selection,

antigen presenting cells (APCs), B-cells, plasma cells and memory B-cells. The algorithm is divided into four phases, namely the initialization, recognition, evolutionary, and response phases. In its initialization phase, self and non-self detectors are generated as well as a population of T helper cells, T suppressor cells and B-cell detectors. The rationale of generating self and non-self detectors is to detect both 'normal' and 'known' anomalies in the data. In phase 2, T-cells, B-cells and APCs perform recognition of the antigen and are activated if the signal is stronger than a threshold value. Phase 3 generates memory B-cells using clonal selection and affinity maturation. Finally, phase 4 generates a primary or secondary immune response depending on initial or subsequent exposure of the antigen, which is determined by the strength of the generated signal. Time series data is used to evaluate the performance of this algorithm. Ji and Dasgupta [105] proposed a real-valued negative selection algorithm that can generate variable-sized non-self detectors. These variable-sized detectors can cover more non-self space as well as avoiding overlapping of generated detectors on self space more efficiently. A similar approach is used in [106], where a negative selection algorithm is developed and applied for an aircraft fault detection system. Shulin *et al.* [107] demonstrated the feasibility of a negative selection algorithm for online fault detection of rotary machinery. Schulze *et al.* [108] extended the ideas presented in [104-106] and applied the algorithm to detecting aerodynamic instabilities in centrifugal compressors to build an early warning system using compressors sound signals.

A hybrid model for immune-inspired network intrusion detection called NetTRIIAD was proposed by Fanelli [109]. Greensmith *et al.* [83] proposed a novel AIS algorithm based on the functionality of DCs as well as using the concepts of danger theory. Zeng and Zeng [110] proposed an AIS anomaly detection algorithm to improve the efficiency of matured detectors by reducing the number of detectors as well as enhancing the coverage of non-self space. The authors used both the innate and adaptive immune systems. In the innate immune system, traditional information security methods are used to generate danger signals, and a population of T-cells are evolved to trigger an immune reaction. This hybrid model is used to achieve high positive prediction values when compared with other similar approaches. Ma *et al.* [111] introduced an antigen feedback mechanism to generate efficient detectors to overcome the problem of random

generation of useful detectors within an acceptable time. A review of the various negative selection algorithms present in the literature can be found in [112].

2.4.2 Optimization

The NIS, by its very nature, does not perform optimization. The goal of any computational optimization is to find an optimal solution. On the other hand, the objective of an NIS is to evolve to find the best possible response (solution) under existing conditions [10], and sometimes it has to work under contradicting conditions. This does not stop AIS researchers using AIS in an optimization domain, however. AIS has produced a number of optimization algorithms on the clonal selection principles such as CLONALG [74], opt-aiNet [113], B-cell algorithm (BCA) [114], and opt-IA [115]. Al-Sheshtawi [116] used CLONALG, opt-IA and BCA on numerical optimization problems and found that the results obtained were comparable with other state-of-the-art optimization techniques. All these approaches use cloning, mutation and selection to build a population of solutions. The authors of aiNet [79] stated some of the benefits AIS can provide while doing optimization: it (a) performs exploration and exploitation of search space through memory cells and antibodies, (b) finds multiple local optimal solutions, (c) maintains many local optimal solutions and (d) has a predefined stopping criteria. An advanced version of opt-aiNet was proposed by Xuhua and called mopt-aiNet, which uses several novel operators such as multi-population, a dynamic hypermutation operator and dynamic memory cell formation [117]. An AIS model for numerical constraint optimization problems based on CLONALG was proposed by Aragon *et al*. [118]. A new mutation operator was introduced that used two different methods for mutation (low and high mutation rates) based on feasibility or non-feasibility of the clones produced. The low mutation of feasible clones helps in exploitation, whereas the high mutation of infeasible clones is useful in exploration of the search space. In recent work, Wang [119] adopted a cluster mechanism to divide a single population into sub-populations for hypermutation and selection. The author also uses a hybrid mutation operator consisting of Gaussian mutation and Cauchy mutation to obtain diversity in antibodies and affinity maturation. This was done to achieve better fitness on the population of antibodies as well as to obtain a global optimal solution. El-Wahed [120] proposed an AIS and neural network hybrid optimization algorithm for finding better solutions near the pareto-optimal frontier. Castro and Zuben introduced a Bayesian artificial immune system optimization algorithm [121] and later upgraded it to

multi-objective optimization (MOBAIS) [122]. The authors replaced traditional cloning and mutation operators with a Bayesian network representing a joint distribution of promising solutions. Another advantage of MOBAIS is its ability to automatically control the population size. The algorithm showed comparative results with other benchmarked multi-objective optimization approaches.

An AIS algorithm inspired by the functionality of T-cells was proposed by Aragon *et al.* [123] for solving optimization problems. This algorithm consists of four components, namely virgin cells (VC), effector cells (CD4 and CD8), and memory cells (MC). The main objective of this algorithm was to explore possible solution space using local and global search operators. An extensive and global search is performed using CD4 and CD8, whereas fine-tuning of candidate memory cells is performed by exploring neighborhood operators (with low mutation rate). An extension of this work for dynamic optimization problems is presented in [124] and called the Dynamic T-cell (DTC) algorithm.

2.4.3 Learning – Clustering/Classification

The earliest AIS algorithm for unsupervised clustering was proposed by De Castro and Zuben [79] and called the artificial immune network (aiNet). It utilizes the concepts of memory cells, clonal selection and hypermutation. This is a two-stage clustering algorithm. In stage 1, a number of memory cells are generated from the original data and then in stage 2 MSTs are used to obtain the number of clusters in the data. Qing *et al.* [125] used a hybrid of aiNet and k-means to perform unsupervised clustering. The authors used aiNet to obtain memory cells which were then used to obtain data partitioning using k-means. And finally, based on obtained data partitioning, the original data was classified.

CLONALG [74, 75] is based on the principles of clonal selection. It was originally designed for engineering optimization but can be adapted for clustering. The algorithm reflects several immune system principles, including the match between antigens and antibodies and the somatic hypermutation and clonal selection of those antibodies that match antigens. Another example of AIS clustering approaches is [126]. Based on CLONALG, this requires a pre-reading of the data to calculate an appropriate threshold for the Euclidean distance metric needed to allocate samples to clusters. The process of clonal selection is also replaced by a random generation of new clusters rather than

adaptation to and specialization for antigens. The concept of genuine evolution of solutions is therefore not fully exploited. A similar AIS algorithm for data clustering was proposed by Liu *et al.* [127], which used the concepts of antibodies and hypermutation to obtain an optimal set of antibodies that can capture antigens (data instances). Timmis *et al.* [89] proposed an AIS algorithm for data analysis which used the concepts of B-cells, hypermutation and network suppression. In their AIS algorithm, B-cell stimulation (learning) is performed at three levels: (1) affinity between antigens and B-cells, (2) affinity of B-cell to neighboring B-cells, and (3) affinity of B-cell to loosely connected neighbors (B-cells).

Knight and Timmis [128] proposed MARIA, an immune system-inspired algorithm for unsupervised learning. This algorithm consists of three layers, namely the free antibodies layer, the B-cells layer and the memory cells layer. It is an iterative algorithm where in each iteration antigens with some probability are selected and exposed to the first two layers (free antibodies layer and B-cells layer respectively). Firstly, an antigen is presented to a free population of antibodies where antibodies are attached to the antigen according to the affinity criteria. Secondly, an antigen and attached antibodies are presented to the B-cells layer. At this stage the following actions can take place: the generation of new memory cells, the generation of a number of free antibodies, or both. If a new memory cell is produced then it is added to the existing memory cell pool, or if new antibodies are generated then they are included into already existing free antibodies. The output of this algorithm is in the form of memory cells, which can be used to cluster unseen data instances. Timmis and Neal (2001) [80] proposed a resource limited AIS approach for unsupervised clustering. The population control mechanism ARBs (mentioned above) is presented to control the exponential growth of B-cells. Nasraoui *et al.* [129] further enhanced the idea of controlling the population of B-cells and proposed a scalable AIS model for unsupervised learning for dynamic and evolving data. The authors proposed an AIS model based on dynamic weighted B-cells and dynamic stimulation and suppression of B-cells.

All the above mentioned algorithms focus only on recognizing the structure (structural feature) of the data and not extracting data partitioning. Li *et al.* [130] proposed an immune system-inspired algorithm for incremental learning called ICAIS. This algorithm generates either of two responses (primary or secondary) when an antigen is

presented. In the secondary response, an antigen is allocated to existing patterns (clusters) whereas in the primary response a new cluster is generated to accommodate new and unseen data patterns. Liu *et al.* [131] proposed a clustering algorithm based on an AIS and a fuzzy system called FAISC and the obtained results on benchmark datasets are claimed to have superior performance than the k-means clustering algorithm.

Hart and Timmis [10] proposed that AISs be incorporated with other biologically inspired techniques, such as neural networks, swarm algorithms and genetic algorithms, to realize their full potential. Rabbani and Panahi [132] proposed a hybrid clustering algorithm based on AIS and bacterial optimization. The algorithm was tested on Iris and Wine datasets. Results were compared against ant colony optimization [6, 25], genetic algorithm [133], tabu search [134], and simulated annealing [5]. It was found that results obtained by an AIS-inspired algorithm were superior to other techniques. Chiu and Lin [135] proposed a hybrid AIS and ant colony optimization algorithm for cluster analysis. Woolley and Milanovic [136] developed a hybrid algorithm based on AIS and a Support Vector Machine (AIS-SVM), to identify voltage collapse-prone areas and overloaded lines in the power system network. In the hybrid approach, the AIS algorithm was used to optimize SVM parameters. The experimental results were compared against simple SVMs. The results obtained using hybrid approach were far superior to simple SVMs in terms of classification accuracy.

2.5 Current State of AIS Research

Hart and Timmis [10] make some suggestions as to the way forward for AIS researchers. The authors believe that AIS systems have had reasonable success in solving the problems of various domains (as mentioned above), but do not provide sufficient advantages over other paradigms. To address current limitations and exploit the available potential of AIS, they identify three key components worth exploring: (1) the innate immune system, (2) the interaction of AIS with other nature-inspired systems and (3) life-long learning.

2.5.1 Innate Immune System

The NIS mainly consists of two sub-systems, namely the innate and adaptive immune systems. AIS research has so far focused mainly on the adaptive side of an NIS due to

its self-evolving, self-sustaining and self-adapting capabilities. Over the years, AIS researchers have dismissed and overlooked the innate immune system and the part it plays in fighting viruses and pathogens, as well as helping to regulate an adaptive immune system. Recently, however, there has been interest in modeling the innate immune system and incorporating it with the adaptive immune response. The work of Greensmith and colleagues [83-85] using DCs to generate danger signals to activate T-cells has identified a promising new direction in AIS research. The full richness of AIS-based algorithms can only be exploited when the adaptive immune system is incorporated with the innate immune system [10].

2.5.2 Interaction of AIS with Other Nature-Inspired Systems

Hart and Timmis [10] proposed that the immune system must be embodied, since natural systems do not work in isolation. It is only through the interaction of the immune system, the neural system and the endocrine system that living organisms can achieve a steady internal state in a dynamically changing environment. The authors suggested integrating AIS with other nature-inspired approaches such as neural networks, swarm algorithms and genetic algorithms to find out the true potential of immune system-based algorithms. However, the way that such integration should occur, taking into account the problem domain, was left open.

2.5.3 Life-Long Learning

One of the prominent features of an NIS is life-long learning. Most of the AIS algorithms proposed to date use static data from standard data repositories and little emphasis is put on building systems which can show life-long learning capabilities. For example, most of the immune system-inspired clustering algorithms proposed in the literature foresee this feature only working in static mode. In summary, Hart and Timmis, outlined a list of features which effective AISs require:

1. “They will be embodied.
2. They will exhibit homeostasis.
3. They will benefit from interactions between innate and adaptive immune systems.
4. They will consists of multiple, heterogeneous interacting, communicating components.

5. Components can be easily and naturally distributed.
6. They will be required to perform life-long learning.” [10]

AIS researchers have noted the comparisons that can be made between machine learning principles and the adaptive immune system, such as antigens which are considered as data samples and antibodies or B-cells which are used to represent data clusters. The interaction of the antigens and the antibodies can be modeled with a normalized Euclidean distance measure. Then random mutation is performed to evolve a population of antibodies. These antibodies provide another representation of the antigens or samples as they evolve to match actual antigens. CLONALG and aiNet reflect all the above-mentioned characteristics. One of the major drawbacks of both aiNet and CLONALG towards unsupervised learning however is their inability to discriminate self patterns from non-self patterns at the learning stage. Put simply, these algorithms do not attach labels (self or non-self) at the learning stage. These approaches need standard statistical techniques to extract meaningful clusters in the data. Therefore, these approaches are considered to only recognize the structural feature of the data in the form of memory cells, and are unable to extract data partitioning on their own. These approaches work in two stages; at the first stage memory cells are generated from data and in second stage clustering is performed on memory cells.

The output of these approaches is in the form of memory cells which can be seen as an exemplar of seen data. According to current approaches, antigens are trapped through antibodies and during this process memory cells are generated. However, the role and functionality of these generated memory cells is not aligned with an NIS. In current AIS approaches, the role of memory cells is static and is only to classify unseen data, whereas in the NIS memory cells are responsible for mounting a faster response (secondary immune response) if the same kind of antigens attack the body. A more efficient and robust AIS algorithm needs to be developed which can incorporate data partitioning capabilities as well as a more active role of artificial memory cells.

In addition to the above-mentioned issues with current AIS approaches, McEwan and Hart [90] evaluated the performance of the AIRS algorithm and core concepts such as clonal selection, affinity maturation and memory cells. The authors demonstrated with experimental results that affinity maturation could be replaced with deterministic operators. They argued that in AIS algorithms, antibodies are the current state of the

system and antigens can be regarded as the final state of the system. Given the current and the final (destination) states, any deterministic approach can produce much faster results than using an AIS operator such as random mutation. Their work removes the need for using a clonal selection process for performing affinity maturation. Most of the algorithms stated in section 2.3 (e.g. CLONALG, aiNet and AIRS) are mainly based on clonal selection process to obtain affinity maturation. The work done in [90] raises the following important questions:

1. Is the clonal selection process for affinity maturation really required in AIS?
2. Can we replace affinity maturation process in AIS with a deterministic approach?
3. Can an AIS algorithm be developed that can remove the need for affinity maturation and still can use clonal selection in novel way?

Given the research questions and motivation of this thesis, and this background review of the research, the remainder of the thesis will demonstrate:

a. A basic humoral-mediated clustering algorithm called the Humoral-mediated Artificial Immune System (HAIS) which shows how biological inspirations can give rise to novel learning algorithms that can be appropriate for variety of benchmarked applications.

b. How this HAIS algorithm, with suitable biologically inspired modifications, can deal with supervised learning.

c. The processes and metaphors that occur at micro level (or cellular level) can inspire an improved learning algorithm. How will these micro level concepts hold together in a macro level framework?

d. Vaccination is an effective learning technique in the NISs. How can we incorporate artificial vaccination in our existing algorithms, and can it provide improved learning capabilities?

2.6 Summary

In section 2.1, the principles of an NIS along with its main organs and cells were discussed, and shown to form the basis of AIS research. Two main processes, namely the innate immune system and the adaptive immune system were discussed, with special

focus on the adaptive immune system. Furthermore, processes such as cell-mediated immunity and humoral-mediated immunity, which constitute the adaptive immune system, were discussed. The components and processes of humoral-mediated immunity such as B-cells, antibodies, negative clonal selection, memory cells, differentiation and proliferation, immune network theory and affinity maturation were also outlined in section 2.1. This section explained vaccination and danger theory phenomena, which are other important learning features of an NIS. Finally, the section ended by listing a number of features of the NIS which are important for AIS and especially machine learning researchers.

The mapping of some of core concepts of the NIS to an AIS was discussed in section 2.2. In section 2.3, a generic framework of an AIS, which consists of representation, affinity measure and immune algorithms, was presented [67]. Then the main AIS-based algorithms (the negative selection algorithm, clonal selection algorithm, immune network algorithm, danger theory, and the AIRS algorithm) were discussed. The inspirations of these algorithms along with some of the limitations presented by each one were also discussed in section 2.3. The main groupings of AIS algorithms based on their application domains in the literature were presented in section 2.4. An extensive review of the work published in these groups was also conducted. The current state of AIS and suggestions as a way forward for future researchers were presented in section 2.5. It is clear from the discussion of Hart and Timmis [10] that AIS has been able to achieve success and obtain good results in a variety of areas, and we believe that AIS has potential to achieve even better results. Some researchers believe that AIS is not bringing novelty to already exiting techniques, while others believe that AIS is one of the ways forward and has a lot to offer. To overcome these issues, Hart and Timmis [10] proposed a list of properties (section 2.3) which an AIS should possess to explore the true potential hidden in AIS research. This list is not a complete list by any means, but definitely can be considered a good starting point. Section 2.5 concluded by highlighting some of the drawbacks in existing AIS machine learning approaches, which will form the basis of this research thesis.

Chapter 3

Humoral-Mediated Artificial Immune System

This chapter presents a novel artificial immune system (AIS) clustering algorithm called the Humoral-Mediated Artificial Immune System (HAIS) that is based on the latest knowledge of the biology underlying natural immune systems (NISs).

The HAIS is inspired by the humoral-mediated response triggered by the adaptive immune system. The key humoral-mediated features of the algorithm include B-cells, antibodies, and memory cells produced through plasma cells. Affinity threshold, network threshold, death threshold, and negative clonal selection threshold are also used to derive intra-cluster and inter-cluster distance metrics that result in the merging of similar clusters, removal/identification of less significant clusters, and segregation of any anomalies in the data. The AIS clustering algorithm proposed here adopts aspects of NISs that have not been previously explored in detail, namely, humoral-mediated responses, as a way of dealing with unsupervised learning as well as the outlier detection problem.

This chapter is divided into two parts. Part I explains the initial HAIS algorithm for unsupervised clustering and part II provides further clarification of the original HAIS algorithm and evaluates its outlier detection capabilities. Section 3.1 briefly explains some of the core NIS concepts employed in developing the HAIS algorithm. These concepts are discussed in more detail in chapter 2 (section 2.1). This is followed by a description of the HAIS algorithm (section 3.2). A detailed explanation of the HAIS algorithm and some of the core parameters are presented in section 3.3. The experimental results on synthetic and benchmark real-world datasets are presented in section 3.4. Section 3.5 concludes part I by describing the novelty and contribution of the HAIS algorithm in the context of existing AIS literature. Part II of this chapter starts with section 3.6, which describes outlier detection in general and its significance in cluster analysis, which is followed by a literature review of existing outlier detection techniques (section 3.7). The proposed HAIS approach and parameter settings for outlier detection are discussed in section 3.8. Section 3.9 presents the experimental results from the HAIS algorithm on simulated and benchmark real-world datasets. Part II and hence chapter 3 concludes with highlighting main characteristics of the HAIS algorithm and directions for further work.

Part I

There is a growing interest in AIS approaches to unsupervised learning. Such interest is the outcome of increased awareness of computational paradigms that can be inspired by natural processes ('nature-inspired computing'). Of particular interest is the way that our body responds to diseases and new pathogens as well as adapting to remain immune for long periods after a disease has been combated. Immune system processes consist of two phases: recognition of invaders or pathogens, and response. The NIS not only provides protection from outside invaders such as viruses and bacteria but also from inside invaders, e.g. cancer cells. Therefore, k=2 clusters that must at least exist in the immune system will be *self* and *non-self.*

An efficient and robust unsupervised AIS algorithm must separate samples/pathogens into different classes based on information available in the sample/pathogen and shared across samples/pathogens. Once such assignments are made, due to the presence of memory of past assignments, when new samples/pathogens enter the system that are similar to previously encountered samples/pathogens it can assign them to the same class/classes as before. One advantage of immune systems is that they respond to brand new pathogens and, for the most part, assign them to new categories to mount the appropriate response. On the other hand, if samples/pathogens are assigned to a new class and later, in the light of new available information (introduction of new samples/pathogens), AIS should be able to re-classify those samples to already existing classes. It is this flexibility and plasticity of immune systems that make them attractive to machine learning researchers.

3.1 Natural Immune System (NIS)

To prepare the ground for introducing the algorithm, we need to describe the human NIS in a way that may not be the most accurate from a bio-molecular perspective but which nevertheless captures the essential features of the NIS, which subsequently inspired our algorithm. The NIS consists of two parts: the innate immune system (IIS) and the acquired/adaptive immune system (AAIS). The IIS is what we have at birth; it remains constant through our lifetime and is our first level of defense. One of the main components of the IIS is leucocytes, or white blood cells, which move through the body

and capture invading micro-organisms and other foreign particles. If the IIS cannot deal with a pathogen, the AAIS is triggered. In what follows, antigens are the equivalent of data samples that are to be clustered.

The main components of the AAIS are also white blood cells, called B-cells and T-cells (lymphocytes). Lymphocytes normally stay in a passive state until they encounter antigens. Each lymphocyte recognizes one type of antigen. Each antigen leaves a genetic blueprint on B or T lymphocytes so that the next time a similar antigen is presented it is recognized quickly by lymphocytes and a fast response mounted. This phenomenon is called immunological memory. If immunity is mediated by T lymphocytes, this is cell-mediated immunity. Humoral immunity, on the other hand, is mediated by secreted antibodies produced in the cells of the B lymphocytes. Secreted antibodies bind to antigens on the surfaces of invading viruses or bacteria and trigger their destruction. The way that antibodies adapt and bind to antigens through a process of 'affinity maturation' (a form of fine-tuning) and 'somatic hypermutation' (a form of constrained search through random mutation) provided the inspiration for the novel clustering algorithm described below.

Very roughly, immature B-cells are produced in the bone marrow and migrate to the spleen, where they are called transitional B-cells which then mature and differentiate into mature B lymphocytes. After maturation, they have a unique B-cell receptor (BCR) on their membrane called an immunoglobulin (Ig) receptor, in the form of an antibody molecule. The BCR recognizes and binds to only one particular antigen. When a B-cell encounters its specific antigen through its Ig receptor and receives additional signals from a helper T-cell (to help prevent the immune system from inadvertently turning on its own body cells), it further differentiates into an effector cell, known as a plasma cell, and a memory cell. Plasma cells, instead of having Ig receptors, start producing and releasing antibodies, which are released to lock onto the antigens, triggering the B-cell into plasma cell differentiation. In other words, Ig receptors have turned into free-floating antibodies through plasma cell versions of the original B-cell.

T-cells are also produced in the bone marrow but mature in the thymus. They too express unique antigen-binding receptors (T-cell receptors or TCRs). As noted above, T-cells are an essential part of cell-mediated immunity. A TCR needs not only an antigen to bind to its receptor but also an antigen attached to a specific major histo-

compatibility complex (MHC) molecule. When a TCR encounters an antigen with an MHC molecule it undergoes division as well as the production of memory cells. Two types of T-cell are T-helper and T-cytotoxic. T-helper cells get activated on contact with antigens and become effector cells [9]. T-cell-mediated immunological response may itself form the basis of a biologically inspired clustering algorithm but this is not the inspiration behind our algorithm, which instead draws inspiration from the B-cell humoral-mediated immunity response described above in approximate detail. The actual human AAIS is in reality more complex than described here. Nevertheless, our approximate description is sufficient to demonstrate a novel approach to clustering.

3.2 Humoral-Mediated Artificial Immune System (HAIS)

An overview of the proposed HAIS is now provided (a full explanation follows the algorithm). HAIS has taken its inspiration from the humoral-mediated immune response triggered in the NIS. A more detailed understanding of humoral immune response can be found in section 2.1.3.2 of chapter 2. Abbreviations used are: B-cell antibodies produced through plasma cell (b-Abs); affinity measure threshold (AT); network threshold (NT); death threshold (DT); memory B-cell antibodies (m-Abs); and negative clonal selection threshold (NST). An antigen is a data sample to be clustered and will consist of specific feature/attribute values. An antibody that 'captures' an antigen will also consist of feature/attribute values that are sufficiently similar to that antigen's values. A B-cell consists of Ig receptors on its surface that are also feature/attribute values. B-cells will form the final set of clusters when the algorithm terminates, with all antigens sticking to at most one B-cell (all data samples falling in at most one cluster).

A. Initialize 1: Create a unique B-cell for every member of an initial set of antigens (Ags) (typically 10% of all Ags) with Ig receptor values matching that antigen.
B. Initialize 2: For each B-cell, produce respective b-Abs, which are similar to, but not identical with, the Ig receptors that matched the antigen.
C. Initialize 3: all the parameters (AT, NT, DT)
D. Repeat
While the pool of Ags is not empty, do:
(1) Label 1: Get an Ag randomly from the pool.
(2) Calculate the affinity measure between Ag and any m-Abs.
(3) ***If*** the similarity measure is less than AT , ***then*** the stimulated m-Ab will do following:
(a) Bring the captured Ag to its respective B-cell, which will hold it (Ag) until the end of this cycle
(b) Go back to Label 1
(4) ***Else***
Label 2: calculate the affinity measure between Ag and b-Abs;
If the similarity measure is less than AT, ***then*** the stimulated b-Ab will do following:
(i) Label 3: Bring the captured Ag to its respective B-cell, which will hold it (Ag) until the end of this cycle

(ii) The B-cell will be stimulated to produce a memory cell with Ig receptors that are an exact match for Ag
(iii) The memory cell will also produce m-Abs with a small random mutation to capture more Ags that are similar
(iv) The B-cell will also be stimulated to make a plasma cell that produces hyper-mutated b-Abs (random evolution)
(v) If any of the newly generated b-Abs is too similar to the existing b-Abs, it will be eliminated through NST and go to Label 4

Else

(vi) Create a new B-cell with an Ig receptor specifically for that Ag and go to Label 3

(5) Label 4: Look for the affinity measure among B-cells (centroids of B-cells – see Step F) and ***if*** two B-cells are more similar than NT, ***then*** the two B-cells stick together to form an inter-connected cluster (Immune Network Theory).

End while

E. If the size of any B-cell is less than DT, remove that particular B-cell (simulating natural T killer cells) and put back the captured Ags into the Ag pool for the next cycle

F. All B-cells will adjust their centroids according to the sub-population of Ags sticking to them. Also, B-cells calculate the local variance of features of captured Ags to update their mutation range for the next generation of Abs. Only a certain percent of Ags closest to the centroids of their B-cell are kept and the remaining Ags will be set free to rejoin the antigen pool for the next cycle

G. M% of the memory B-cells will go into the next cycle (the M% nearest to the centroids of the surviving B-cells)

H. All b-Abs are killed and a new population of Abs generated from the surviving B-cells.

I. Update all the thresholds parameters (AT, NT, DT) for the next cycle

J. **Until** termination condition.

An overview of the HAIS algorithm is provided in Figure 3.1 below and a description of the stages follows: (1) Antigens are released from the antigen pool. (2) Some antigens stick to the Ig receptors in the surface of B-cells, which then differentiate into plasma cells and memory B-cells. (3) Plasma cells produce and release antibodies that are (a) identical to the Ig receptors of their parent B-cell so that more identical antigens are captured, and (b) slightly different so that almost identical antigens are captured. If any of the mutated antibodies are too similar to the original antibodies, they are eliminated through negative selection. (4) Memory cells have receptors that are both identical to their parent B-cell receptors to capture identical antigens and slightly mutated receptors to capture slightly different antigens. (5) If an antigen sticks to no B-cell or antibody, a new B-cell is generated with Ig receptors specifically for capturing that antigen. (6) If B-cells do not attract a sufficient number of antigens (death threshold) through their receptors, they are destroyed by natural T-killer cells and their antigens released for recirculation back to the antigen pool. (7) All B-cells adjust their centroids depending on the antigens sticking to their receptors. When two or more B-cells have centroids that are closer than a network theory threshold, they are merged into one B-cell.

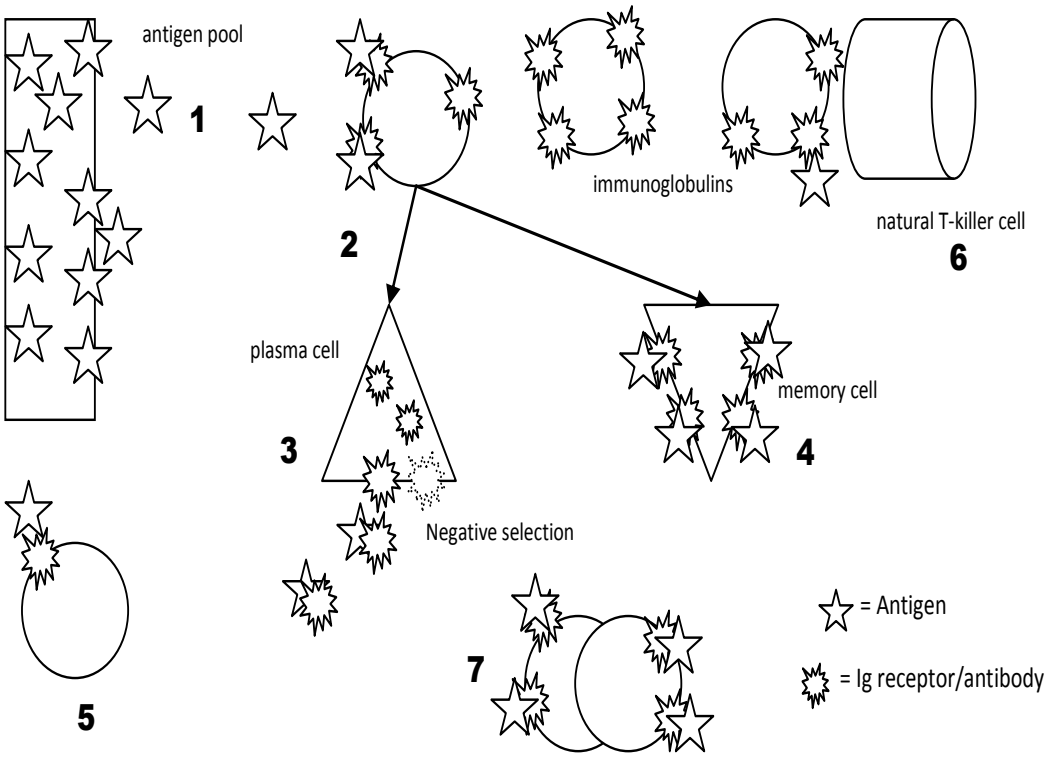


Figure 3.1: Humoral-mediated B-cell immunity, from an organizational perspective

3.3 HAIS Algorithm Explanation

The mapping of AIS expressions to classical clustering concepts is shown in Table 3.1.

The algorithm starts with generating a small number of B-cells from the initially and randomly presented antigens (Step A). No pre-reading of the entire set of antigens is required to calculate parameter values. This initial number of B-cells can be any reasonable number higher than zero and less than the number of total antigens present in the antigen pool. Our experiments so far indicate that a random 10% of the dataset is sufficient as a reasonable initial number of B-cells for successful clustering of all samples in the dataset. This figure of 10% is used throughout the experiments reported here (unless otherwise stated) as a fixed point in the experimental method. The similarity measure between an antigen and Ig or antibody (Step D.3) is calculated through squared Euclidean distance:

$$D = \frac{1}{c} \sum_{i=1}^{c} (A_i - B_i)^2 \qquad (1)$$

where D is the similarity measure, c is the number of features (number of columns in the data if the antigens are stored as rows), and A_i and B_i are the Ig/antibody and antigen features respectively.

Table 3.1: Mapping of AIS expressions to classical clustering concepts

NIS Expressions	Clustering Concepts
Antigens	Samples
B-cells	Clusters
Antibodies	Mutated copies of captured samples
Interaction between antibodies and antigens	Pair-wised comparison
Similarity measure	Normalized Euclidean distance
Mutation	Creation of diverse population of solutions
Memory cells	Already known patterns
Affinity threshold	Similarity criterion

If the minimum of the similarity measure is less than the affinity measure threshold (AT), the closest matched Ig will be stimulated. But if the similarity measure is more than the AT then a new B-cell and associated Ig are generated to capture the loose antigen (Step D.4.vi). In the process of generating new antibodies, the process of negative selection is used to increase the diversity of antibodies in the system. Each newly generated antibody is checked against all the existing antibodies, and if the newly generated Ab is too similar to any of the existing ones then the newly generated Ab is eliminated (Step D.4.v). This negative selection (similarity measure) is controlled by a parameter NST, which can be changed.

AT and Network Threshold (NT) start with the same value but the former decreases as the number of cycles increases whereas the latter increases with the increase in the number of cycles (Step I). The direct and inverse proportion of these parameters with the number of cycles is designed to obtain convergence. As the number of cycles increases, the algorithm discourages the formation of new clusters as well as the merger of already existing clusters. This is to allow the AIS to settle on a stable number of clusters (convergence). Below are the two equations used to set the initial AT and NT values based on the initial selection of antigens:

$$AT = NT = \frac{\frac{1}{c}\sum_{i=1}^{c}\sigma_i}{\alpha} \quad (2)$$

$$AT = NT = \frac{\frac{1}{c}\sum_{i=1}^{c}\sigma_i^2}{\alpha} \quad (3)$$

In (2), c is the number of features in the data and σ is the standard deviation of each feature. Equation 2 is useful when the data is normalized whereas (3) is useful when the data is not normalized. The parameter α is a scalar value, which controls the tightness of boundaries among the clusters. AT and NT are updated as follows:

$$NT = NT.nt_{rate} \quad where \quad nt_{rate} \leq 1 \quad (4)$$

$$AT = \beta.new_{aff} + \gamma.Bcell_{aff} \quad (5)$$

$$where \quad \beta + \gamma = 1$$

$$new_{aff} = AT.at_{rate} \quad where\ at_{rate} \geq 1 \quad (6)$$

$$Bcell_{aff_j} = \left(\frac{1}{f}\sum_{i=1}^{f}\frac{\sigma_i(Bcell_j).L}{\alpha}\right).at_{rate} \quad (7)$$

where L is the number of clusters.

Each B-cell (cluster) is evolved according to the parameters β and γ (Equation 5). If β is set to 1, the affinity measure for each cluster will be updated equally. If γ is set to 1, each cluster is updated separately depending on the instances it captures at the end of each cycle. In the case of 0.5 for both β and γ, the AT update will consider both as static increments as well as increments based on the clustering obtained at the end of each cycle. If the user is not sure, 0.5 is recommended for both parameters. The parameter values β and γ can be changed appropriately in the light of any prior knowledge available regarding the structure of data. If β is set to 1, the HAIS algorithm might not be able to detect outliers as each cluster will increases its AT at the same rate. But if γ is set to 1, then larger-sized clusters can evolve more rapidly, thereby discouraging natural competition among clusters (B-cells). The effects of β and γ will be discussed in the experimental section.

Death threshold (DT) is used to encourage competition in the immune system where each existing cluster (B-cell) fights for its survival and, if at the end of the cycle, a cluster (cluster size) is less than the threshold value, it is eliminated. DT is directly proportional to the number of cycles: as the number of cycles increase, so does the DT

parameter. DT is bounded between X and Y positive integers, $X \leq DT \geq Y$. The DT parameter is used to capture outliers in the data. Negative selection threshold (NST) is used to increase the diversity among antibodies. This parameter is set so that there is enough diversity present at all the times in the system. Setting this value too high will put too much survival pressure on the newly generated antibodies. Setting it too low will make the size of the antibody population too high in the system, resulting in the algorithm becoming computationally expensive.

Memory cells are more specialized for specific kinds of antigen. When comparing the similarity measure between antigens and memory cells through memory cell receptors, the memory cells need a higher affinity to get stimulated as a response to the antigen. For the experiments undertaken here, antigen to memory cell Ig affinity is set 10 times higher than the antigen to antibody affinity. At the end of each cycle only a specified percentage of the memory cells will go into the next cycle, and that ratio is set to 40% to provide direction and momentum to the clustering. At the end of each cycle, each B-cell calculates its centroid and, for the next generation, it will keep k-nearest neighbor (k-NN) of antigens closest to the centroid, with the remainder sent back to the antigen pool. This k-value can be static or dynamic (increasing with the increase in number of cycles). Here, the value is kept static at 1-NN from the centroid of each B-cell. In other words, only one nearest neighbor is kept at the end of each iteration. Memory cells will eventually represent the essential similarity between the numbers of samples that belong to a cluster.

Mutation is based on the directed evolution principles of [137], where B-cells adapt their Ig and Ab production mechanisms for generating features that reflect the local variation found in the antigens collected so far. Different features can have the same mutation rate or all the features in the data can have different mutation rates. Mutation is designed in such a way that each selected feature can mutate both ways (positively and negatively) at random, rather than just in one direction. While the mutation is constrained by the variance in the collected antigens, the choice of feature to mutate is random within B-cells. If the mutation rate is too high the clusters may start to overlap, which results in cluster over-fitting. On the other hand, if the mutation rate is set too low the clusters may under-fit.

3.4 Experimental Results

A two-part experimental technique was adopted. In the first part, three synthetic datasets were produced with varying numbers of clusters. The aim of the first part of the experiments was to test whether the HAIS algorithm could find clusters known to exist in the synthetic data. The second part of the experiments describes the results obtained when the HAIS clustering algorithm was run a number of times on datasets that are commonly used for testing the behavior of new clustering algorithms. The results of the HAIS algorithm are compared with classical (i.e. non-stochastic) results where appropriate to determine whether the new algorithm offers any advantages over standard clustering techniques. Three benchmark datasets, namely Iris, Breast Cancer and Network Intrusion, are used. More details regarding these datasets can be found in appendix A. The HAIS algorithm was executed in MATLAB and all experiments performed on an Intel i7 CPU 2.93GHz with 3 GB RAM.

3.4.1 Synthetic Data 1

The data consist of four well-separated clusters with clearly defined cluster boundaries. Each cluster consists of 50 samples/antigens (total 200 instances), each consisting of two features (Figure 3.3 [L]). The HAIS algorithm converged to the correct clusters within 20 presentations of the antigens (Figure 3.2). The final instance-wised classification of synthetic data 1 can be seen in the Figure 3.2 (R) which shows that the algorithm has been able to classify all the samples accurately. The process was repeated a further 9 times with different random initial sample sets, with correct convergence each time.

The 2-D projection of the synthetic data 1 (four clusters) can be seen in Figure 3.3 (L) where each cluster is identified separately with unique color and shape. As noted earlier, memory cells constitute a distinguishing feature of humoral-mediated immunity and in the HAIS algorithm perform the role of data summarization. The memory cells obtained by the algorithm are shown in Figure 3.3 (R) and can be used for clustering new samples as they appear. Memory cells represent all four clusters with fewer samples and also keep intact the original shape and structure. The mutation rate was set to 5% and alpha to 1 for this dataset.

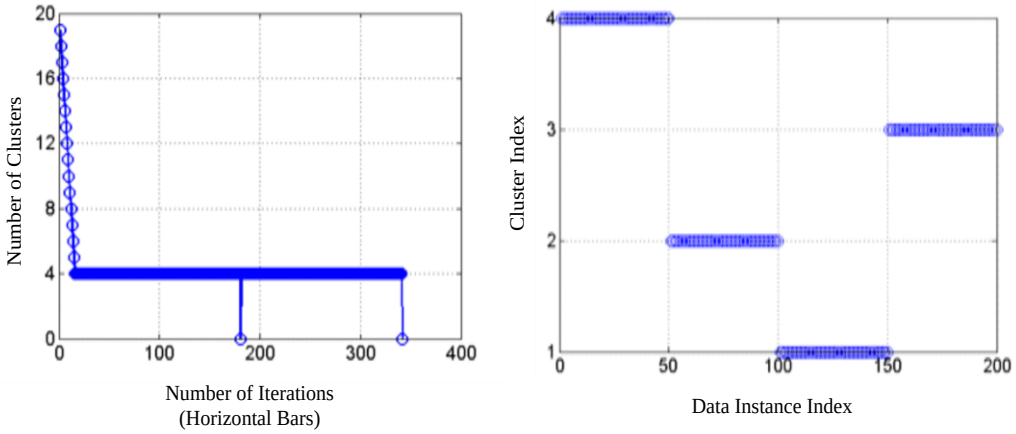


Figure 3.2: L: Cluster formation (y-axis) against samples (x-axis, 20 antigens/samples initially presented, 180 left in pool), showing that within two cycles (vertical bars) the clustering had converged, satisfying the termination condition of no change in two consecutive cycles (vertical bars on x-axis). R: Classification (test of accuracy of clustering) showing the instance-wise classification of the synthetic data, with all instances correctly classified (no overlap of instances on the x-axis) against their clusters (y-axis).

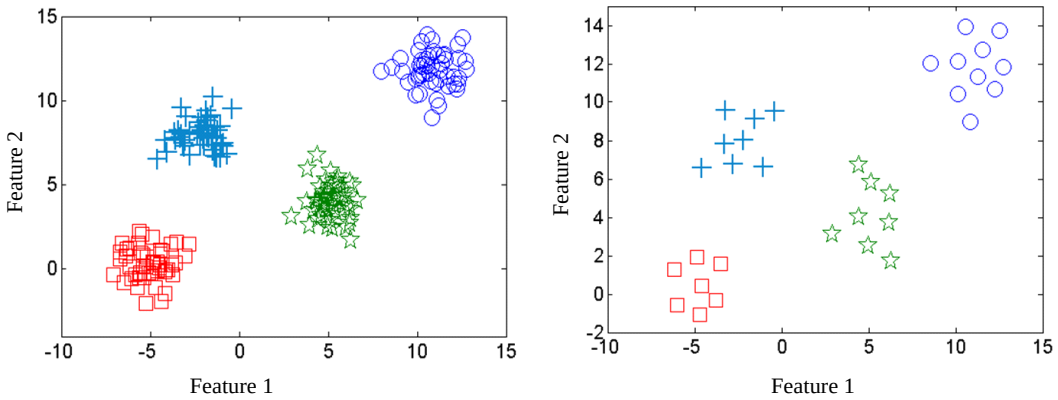


Figure 3.3: L: Clustering obtained of synthetic data 1, showing clear separation on the two features. R: Final set of memory cells obtained for synthetic data 1, showing data capture and reduction.

3.4.2 Synthetic Data 2

Synthetic data 2 has 1000 instances, with two features forming 14 clusters. In this particular case, because of the large number of antigens, 10% of the initial antigens were used to calculate the AT and MT and only 10 B-cells initially produced. Also, for this dataset the number of instances in each cluster and size of each cluster are different (ranging from 60-80 per cluster). All the clusters are slightly elongated in shape. The 2-D projection of the data can be seen in Figure 3.4 (L). The HAIS algorithm finds the correct number of clusters in the first cycle after an initial growth to 16 clusters. The

final number of clusters obtained by the algorithm can be seen in the Figure 3.4 (R) and demonstrates that the algorithm both merges and generates clusters as needed while still in the process of analyzing the samples as they enter the system (Figure 3.4 [R], cycle 1).

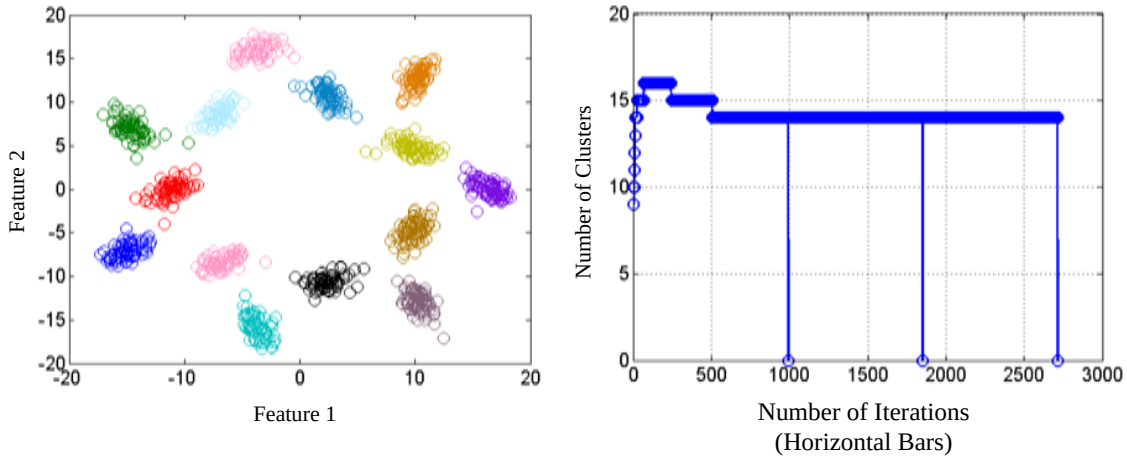


Figure 3.4: L: 2-D projection of the clustering obtained, using different colors as clusters (14 clusters). R: Instance-wised cluster formation for synthetic data 2, showing three cycles.

3.4.3 Synthetic Data 3

The data consist of three well-separated clusters with clearly defined cluster boundaries along with some outliers consisting of a cluster-outlier and four single-point outliers (Figure 3.5). Each cluster consists of varying samples/antigens, each consisting of two features. The nt_{rate} and at_{rate} (Equation 4 and 6) are set at 0.90 and 1.10, respectively. Mutation rate is set at 5% for both features, DT is 15 and β and γ are set at 0.25 and 0.75 respectively. The final instance-wised cluster formation of synthetic data 3 can be seen in Figure 3.6 (L) which shows that the HAIS algorithm has been able to find three main clusters. As can be seen in Figure 3.6 (R) the algorithm does not converge due to the presence of outliers. At the end of each cycle the algorithm finds 8 clusters but, due to the setting of DT at 15, the remaining clusters die out and the algorithm finds three clusters at the end of each cycle. The HAIS algorithm was run for 6 cycles with each cycle showing the same behavior.

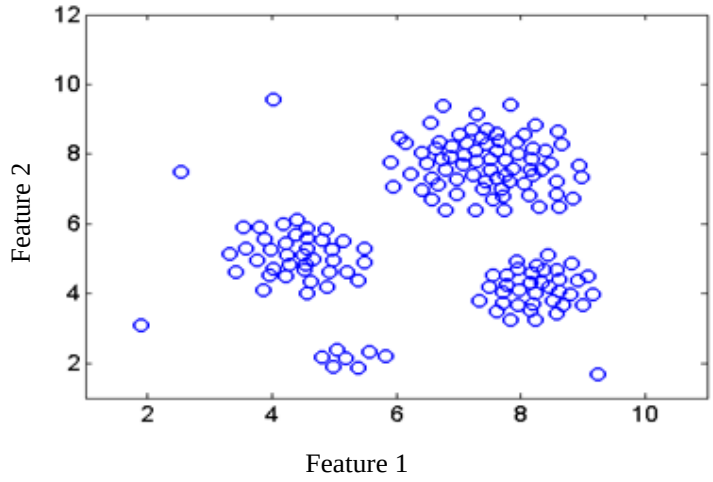


Figure 3.5: Projection of synthetic data 3, showing various clusters as well as outliers

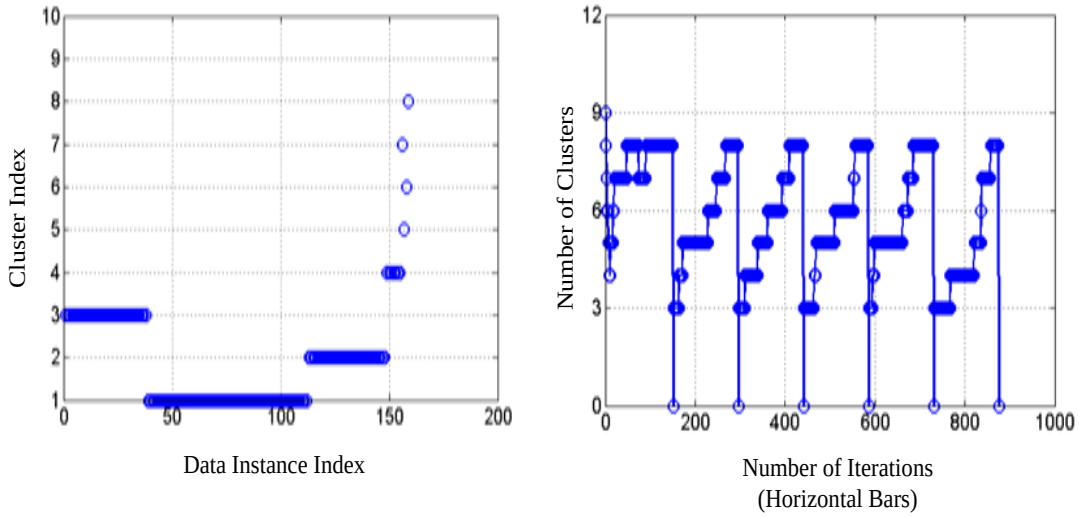


Figure 3.6: L: Clustering (test of accuracy of clustering) showing the instance-wise class association of the synthetic data 3, with three main clusters and remaining outliers. R: Instance-wised cluster formation for the synthetic data 3, for six cycles.

Synthetic data 3 was tested for different values for β and γ. Table 3.2 presents the B-cell (cluster) AT at the end of each cycle when β and γ are set at 0.25 and 0.75 respectively. Each B-cell starts with the same AT value but gradually converges on its local AT value, depending on the antigens trapped by the respective B-cell. These β and γ parameters are used to assign each cluster its own AT parameter value as real-world clusters vary in sizes and shapes. It can be seen in Figure 3.6 (L) that the HAIS algorithm was able to find 3 clusters and 5 outlier clusters in the first cycle, but the

algorithm was run for another 5 cycles to demonstrate the non-converging and interesting oscillatory behavior of the algorithm in the presence of outliers in the data.

Table 3.2: AT parameter at the end of each cycle with $\beta = 0.25$ and $\gamma = 0.75$

Index	Cluster 1	Cluster 2	Cluster 3
Cycle 1	0.6038	0.6038	0.6038
Cycle 2	0.7634	0.6414	0.7207
Cycle 3	0.9438	0.6531	0.7572
Cycle 4	1.0002	0.6568	0.7687
Cycle 5	1.0178	0.6580	0.7722
Cycle 6	1.0233	0.6583	0.7733

On the other hand, when β and γ are set at 1.0 and 0.0 respectively, the B-cell affinity values found at the end of each cycle can be observed in Table 3.3. By using these parameters for β and γ, the AT for each cluster increases with the same rate after every cycle.

Table 3.3: AT parameter at the end of each cycle with $\beta = 1.0$ and $\gamma = 0.0$

Index	Cluster 1	Cluster 2	Cluster 3
Cycle 1	0.6038	0.6038	0.6038
Cycle 2	0.7547	0.7547	0.7547
Cycle 3	0.9434	0.9434	0.9434
Cycle 4	1.1793	1.1793	1.1793
Cycle 5	1.4741	1.4741	1.4741
Cycle 6	1.8426	1.8426	1.8426

In conclusion, running the HAIS algorithm on synthetic data demonstrates the feasibility and effectiveness of the humoral-mediated clustering approach. The next task is to evaluate the effectiveness of the algorithm on well-known datasets that have been extensively studied in the past.

3.4.4 Iris Data

When the HAIS algorithm is run on Iris data, the number of clusters in each cycle and also instance-wised cluster formation is shown in Figure 3.7. At the end of the first cycle, the HAIS algorithm found four clusters in the data. Cluster 4 was below the

threshold (DT) value and removed by the algorithm before the start of the next cycle. The second cycle started with three clusters and the algorithm converged after one more cycle. Equation 2 was used for this dataset.

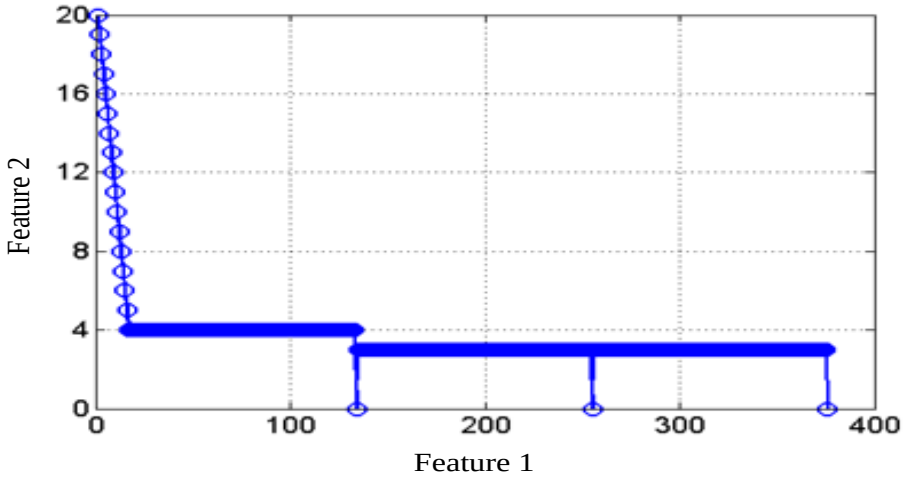


Figure 3.7: Instance-wised cluster formation for the Iris data for three cycles

The clustering results based on each instance are shown in Figure 3.8 (L). There are 14 errors in this case. Clustering is dependent on the order of presentation of instances and also on the initial B-cells selected. The final 2-D projections of the clusters from the HAIS algorithm are shown in Figure 3.8 (R). The algorithm works better than classical 2-step clustering (automatic clustering using the Bayesian Information Criterion (BIC): 20 errors) and k-means (k=3) using Euclidean distance (16 errors). Standard hierarchical clustering (agglomerative) [138] produced 15 errors. Different runs of our algorithm can produce different cluster formations and also number of clusters due to the initial random 10% of samples and order of instance presentation. Nevertheless, the algorithm found the correct number of clusters (and correct allocation of samples to clusters) 7 times in 10 separate runs. On a majority vote criterion (7 out of 10), the three-cluster solution wins over other cluster solutions for the Iris data using the HAIS algorithm.

To demonstrate the stochastic behavior of the HAIS algorithm, 50 runs of HAIS algorithm using the Iris data were performed and the results (clustering errors) are shown in Figure 3.9, which shows oscillation between good and bad clustering results. The average outcome of these 50 runs is 16.46 errors. The best error obtained was 7, which was at run 12, and the worst error was at run 2, which was 27.

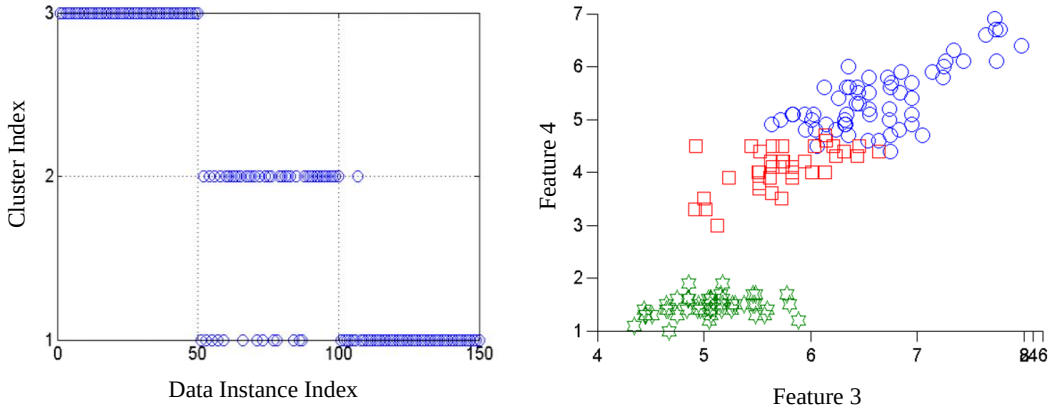


Figure 3.8: L: Instance-wised classification of the Iris data showing 14 errors. Misclustering of samples in one class is represented by circles in the vertical column of another class. The figure shows that 13 samples of cluster 2 fall in cluster 1 and 1 sample of cluster 1 falls in cluster 2. R: 2-D projection of the clustering obtained for the Iris data using different shapes and colors as clusters

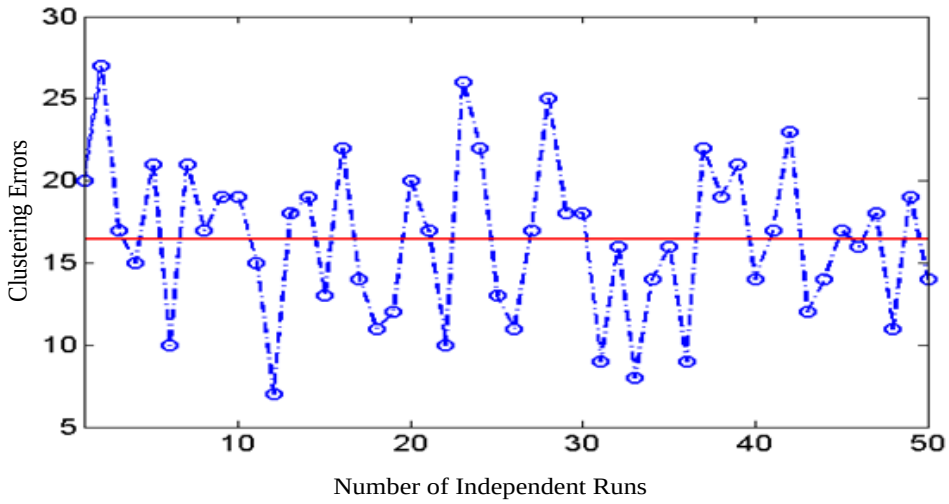


Figure 3.9: HAIS clustering algorithm, final clustering results (3 clusters) for Iris data. The x-axis represents number of runs while the y-axis shows number of errors against true class labels.

3.4.5 Breast Cancer Wisconsin

This dataset was normalized between -1 and +1, and Equation 3 used with alpha set at 3. This dataset was selected to demonstrate that a higher number of features does not affect the outcome of the HAIS clustering algorithm. The HAIS algorithm started with 20 randomly selected data instances and converged to the right number of clusters after six cycles (Figure 3.10). The algorithm initially found 14 clusters in the data but the DT forced most of the clusters to be eliminated by the end of cycle 3. The algorithm finally converged to two clusters after the fourth cycle.

The HAIS algorithm was run 10 times. One of the final classifications obtained is shown in the Figure 3.11 (L) where the total number of misclassification was 36 out of 569. The distribution of objects with respect to each cluster at the end of first cycle can be seen in Figure 3.11 (R) showing that at the end of the first cycle the algorithm found two major clusters in the data, with the remaining clusters being eliminated through the DT. The breast cancer dataset was also clustered using standard k-means (k=2) and 36 classification errors were also returned. When the data was analyzed with hierarchical model based clustering [138], 68 errors were returned. Hence, the HAIS algorithm appears to be no worse than existing clustering algorithms when performing at its best for this dataset.

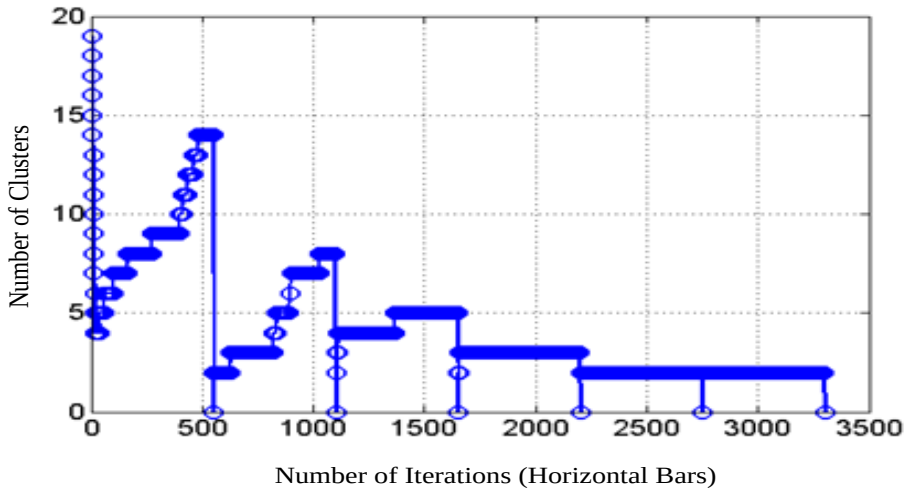


Figure 3.10: Instance-wise clustering formation for the Breast Cancer Wisconsin data for 6 cycles

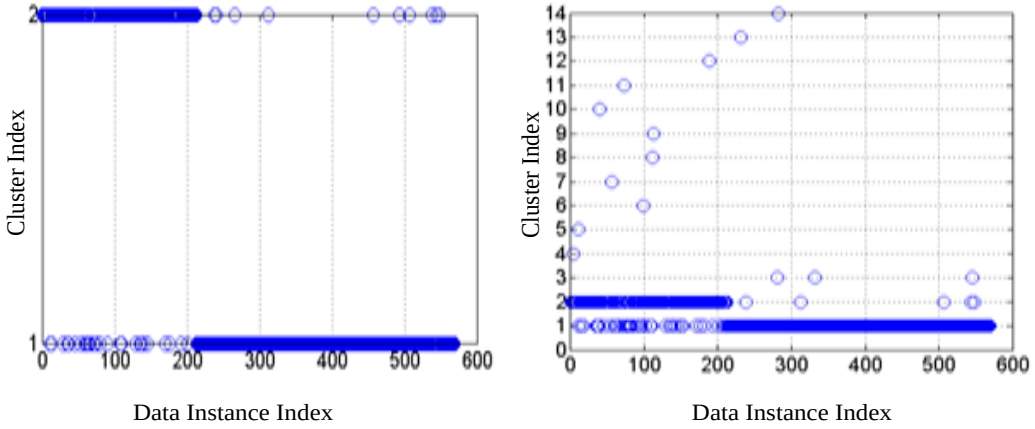


Figure 3.11: L: Instance-wised classification of the Breast Cancer Wisconsin data at the end of a run, showing 36 errors. R: Instance-wised clustering at the end of the first cycle for the Breast Cancer Wisconsin dataset showing 14 clusters.

3.4.6 KDD Intrusion Detection Dataset

So far we have demonstrated through experiments that the HAIS has the capability of finding natural groupings in data where datasets have variable spatial dimensions. However, the number of instances in all these datasets were not more than a few hundred. The KDD intrusion detection dataset is used to demonstrate that the HAIS can work effectively with large-scale datasets. This dataset has 5 classes but in this experiment the HAIS algorithm was used on two classes only: Normal and DOS. Random 10% samples of the Normal and DOS classes, resulting in 9,728 and 39,146 instances respectively, were selected for the experiment. The continuous features were normalized (between -1 and +1). For continuous features a 4.5% mutation range was used, whereas categorical data instances were mutated by randomly flipping one bit. The DOS super-class is divided into six classes: Back, Land, Neptune, Pod, Smurf and Teardrop. More detail about this dataset can be found in [139].

The original projection of both Normal and DOS class objects is shown in Figure 3.12 (L) where spherical-blue objects represent the normal class and triangle-red objects the DOS class. Figure 3.12 (R) shows the 4 clusters found by the HAIS algorithm after one typical run.

The Normal instances (dark blue in Figure 3.12 [L]) are nearly always captured by one cluster (Group 1) across all runs, whereas DOS instances are typically sub-clustered into three groups of different sizes. The confusion matrix obtained by the HAIS algorithm for this run is shown below, where the cluster index shows the true clusters labels. The group index shows the three sub-clusters of the mostly DOS instances found by the algorithm.

	Group1	Group2	Group3	Group4
Cluster 1	9280	6	4	438
Cluster 2	243	8710	28189	2004

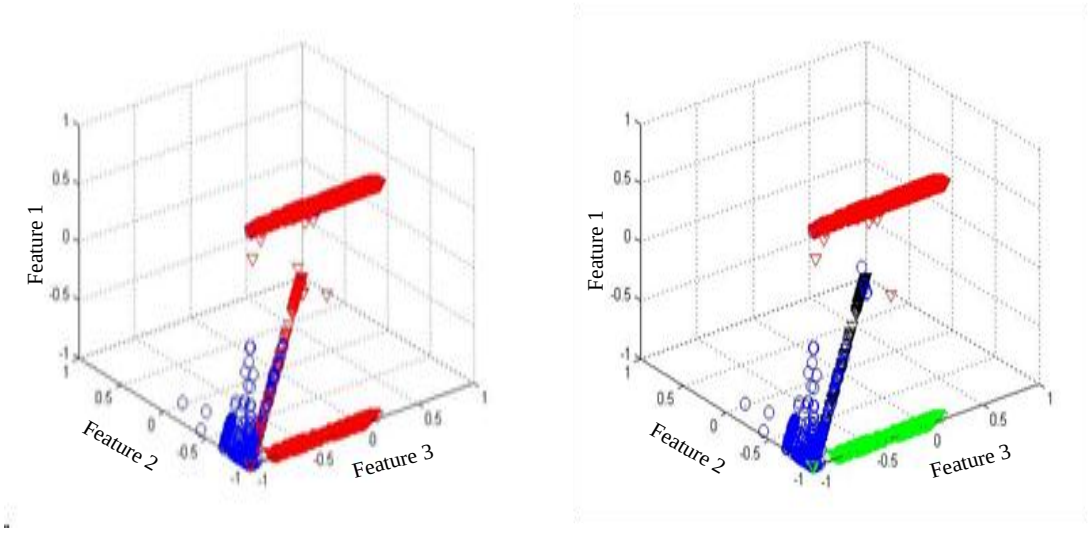


Figure 3.12: L: Original two projections of the Normal and DOS classes of the KDD dataset. R: Final obtained clustering using HAIS algorithm showing 4 clusters, using different colors and shapes.

These results can be simplified using the 'Normal/Non-normal' binary (i.e. Group 1 vs. the rest) to:

	Group1	Group2
Cluster 1	9280	448
Cluster 2	243	38903

Accuracy ((TP+TN)/(TP+FP+TN+FN), where 'T'=true, 'F'=false, 'P'=positive and 'N'=negative) across 10 runs on a binary clustering metric (i.e. all non-normal are counted as DOS) is 98.18%, with sensitivity (TP/(TP+FN)) of 94.37% and specificity (TN/(TN+FP)) of 99.12%.

When re-running the data using standard (and deterministic) k-means clustering (setting k to first 2 then 4), the confusion matrix for k=2 is:

	Group1	Group2
Cluster 1	9562	166
Cluster 2	11025	28121

The overall accuracy of 77%, sensitivity of 98.2%, and specificity of 71.9% are due to the large number of DOS instances classified as Normal. For k=4 means analysis, the confusion matrix shows a better grouping and fewer classification errors:

	Group1	Group2	Group3	Group4
Cluster 1	9197	3	4	524
Cluster 2	353	8683	28107	2003

Accuracy is 98.19%, sensitivity 94.54% and specificity 99.55%. These figures (single run) are almost identical to the overall figures returned by the (stochastic) HAIS algorithm over 10 runs. However, the main difference is that the HAIS algorithm is strictly unsupervised (no class or cluster information of any sort is entered into the system), whereas k-means analysis requires the user to enter a desired or expected number of clusters.

In comparison to previous clustering approaches to this difficult dataset, our AIS is by no means the worst. For instance, Gaussian classifiers, incremental radial basis function, and fuzzy adaptive resonance theory, among others, return sensitivity figures ranging from 73% to 97.3% and specificity figures ranging from 99% to 99.7% (all for Normal vs. Dos only) [139]. The HAIS algorithm sensitivity is therefore at the high end and its specificity in the middle range of what has been published previously.

3.5 Summary

We have described a novel humoral-inspired artificial immune system-based clustering algorithm (HAIS) inspired by memory cells, plasma cells, Ig receptors and antibodies. In contrast to other approaches [74, 79, 126, 131], the HAIS algorithm requires no pre-reading of the data. The experimental results indicate that dynamic growth and reduction of the number of clusters within and across cycles, together with merging of clusters, achieve effective clustering. The methods for merging and removing clusters through humoral-mediated techniques are novel. Results on synthetic and real-world data demonstrate the feasibility of the approach when clusters are known to exist in the data. In synthetic dataset 3 we have demonstrated that the HAIS has the capability to identify clustering and outliers in the data. Later in this chapter we will focus on extending the algorithm further to deal with more complex outlier detection problems. Also, future emphasis will be on constructing a better dynamic framework for reinforced memory transfer between cycles. The HAIS algorithm was originally devised to achieve continuous or online learning; so far its performance has only been tested on static datasets.

The HAIS algorithm is similar in many respects to other AIS clustering algorithms in the literature that use B-cells and memory cells. However, the HAIS algorithm does not allow direct mutation of antibodies. Mutation is instead through the B-cells that produce the antibodies. Also, our AIS includes an explicit reference to plasma cells during the clustering process as well as Ig receptors for matching. Do these differences matter?

It has been almost 9 years since the review by Dasgupta *et al*. [9], where it was noted that AIS systems at that time used immune systems 'piecemeal' due to the bio-molecular complexity of what was then known of the NIS. In particular, the reliance on just three immunological principles (immune network theory, negative selection, and clonal selection) was identified. That review concluded with recommendations on how an AIS could be made more useful as a problem-solving technique, including enhancement of representation and the introduction of other mechanisms as necessary.

The HAIS explicitly incorporates the process of antibodies mutating not by themselves but within plasma cells that then release the antibodies to capture antigens. In other words, our algorithm is more faithful to the principle of new antigens not previously encountered being matched against Ig receptors on the surface of B-cells. When antigens stick to the Igs, the B-cell evolves into a plasma cell, which releases antibodies that are both faithful copies of the original Igs as well as mutated versions of those Igs. Also, the original, triggered B-cell makes a memory cell so that if the same antigen reappears in the future, it can be more quickly captured by the system. Our algorithm also uses the concept of natural killer cells to remove any B-cells that are not activated by antigens. The HAIS therefore introduces new mechanisms not previously used in AIS algorithms as well as novel representations of Igs and antibodies.

There are computational advantages in being more faithful to the biology. For computational antibodies to mutate directly means that every antibody must now contain the evolutionary mechanisms required for mutation as well as building the modified receptors. This is redundancy on a grand scale, especially if the antibodies are not successful at capturing antigens and must therefore be destroyed. It is conceptually more elegant and efficient to stick with what we know about how the NIS produces antibodies, which is through B-cells and their plasma equivalents. These cells now contain the necessary mutation and bio-molecular machinery to produce antibodies of a possibly infinite variety. On the other hand, being more faithful to biology brings with it

a few disadvantages: (1) it brings possibly unnecessary algorithmic complexity by adding biological details; (2) a reasonably good understanding of biological processes is required to deeply understand the algorithm; and (3) computing algorithms can only establish an abstract level of mapping with true biological processes.

Part II

In section 3.4.3, a synthetic dataset consisting of three distinct clusters as well as some outliers was used to depict the behavior of the HAIS algorithm in the presence of outliers in the data (see Figure 3.5). Due to the presence of outliers in the dataset, the algorithm did not converge to a stable number of clusters (Figure 3.6 [R]). This non-convergence is in contrast to the normal HAIS behavior that converges on a stable number of clusters after a number of iterations. In the experiment conducted in section 3.4.3, at the end of each iteration the HAIS algorithm found 8 clusters, out of which three were normal clusters and due to the implementation of DT parameter, the 5 clusters having fewer instances were removed. Therefore, after applying the DT parameter the HAIS algorithm produced three clusters. This behavior can be observed in Figure 3.5 (R). It was concluded from the experiment discussed in section 3.4.3 that the HAIS algorithm has the capability to find natural groupings as well as outliers in the data simultaneously. In other words, the presence of outliers does not affect the capability of the HAIS algorithm of finding natural groups in the data. Here, in part II, this oscillatory (non-converging) behavior of the HAIS (illustrated in Figure 3.6) is further explored using synthetic and real-world benchmark datasets for dealing with outliers.

3.6 Outlier Detection

Data mining has been used extensively for discovering hidden or implicit patterns in data. Clustering is an exploratory data analysis technique that involves grouping data instances together in such a way that members within each group have maximum within-group similarity and groups have maximum between-group dissimilarity. Identifying natural groupings of patterns in the data is one of the key aims of any data clustering technique. However, in some real-world applications, it is also important to find outliers in the data. Outliers are objects in the data which are significantly different from the rest of the data. Outlier detection is the process of finding such anomalies in the data. It has many real-world applications, such as detecting fraud or criminal activities in e-business, identifying potential attacks on computer systems through intrusion detection systems and, generally, finding interesting or significant exceptions in the data. An example of an exception in the area of performance evaluation is the

issue of how to identify a very small number of underperforming employees in a large employee database in such a way that such cases are clearly identified as separate from other cases.

In unsupervised learning, outliers are typically considered as noise and, because they can severely affect the results of clustering, are removed from analysis. This is because of the fundamental clustering principle, which is that objects within a group/cluster should have more in common with each other than any single object in a group has with any other object in another group. An outlier, by its very definition, is an object that has very little in common with any other object. In clustering, such outliers are usually given a cluster all to themselves because of their distance or separation from other clusters. This can significantly affect the way that other 'normal' (non-outlier) objects are clustered, since normal samples will now be forced into a cluster with each other due to the distance they all have from the outlier cluster. Subtle differences between normal objects, which would otherwise lead to separate clusters, will be dwarfed by the large distance from the outlier cluster that they all share. Since one of the aims of clustering can also be to find the minimal number of clusters in the data, the presence of one extreme outlier out of 100 samples could lead to all other 99 samples being located in just one cluster. Removing the outlier to enable the other 99 samples to be properly clustered could lead to the identification of yet another outlier that causes the same problem for the remaining 98. While the distances between outliers and 'normal' clusters will reduce with each iteration, there is currently very little understanding of when to stop the outlier removal process. In addition, there is a danger that continuation of the outlier removal will lead to not just noisy cases being excluded but also 'interesting' cases that are genuine outliers being lost. The adoption of arbitrary thresholds for the removal of outliers can also raise questions about the validity of the clustering. In other words, there is currently very little understanding in clustering of how to separate noisy and therefore faulty data from genuine outliers that are not consistent with the underlying patterns in the data.

Outliers can exist in the form of single point or as a small cluster [140]. Outlier detection approaches can mainly be divided into four categories: distribution-based, distance-based, density-based, and cluster-based approaches [141, 142]. Outlier detection is assumed to be a side-product of clustering methods [143-145]. Overall, current approaches are characterized by the use of post-filtering of samples and

repetition of clustering with ‘noisy’ or ‘erroneous’ data removed. The aim of this work is to explore an alternative approach where outliers and normal clustering can take place together without the need for post-filtering. In other words, whereas the approach of current outlier detection techniques is to regard outliers as noisy and faulty until proven otherwise, the aim of our work here is to explore an alternative approach, where outliers are regarded as genuine until proven otherwise. The main problem for this alternative approach is to find ways of allowing normal clustering to occur in the presence of outliers and without post-filtering.

As noted earlier, the task of any clustering algorithm is to find natural clusters and patterns in the data, but in the presence of outliers clustering algorithms show poor results. Most of the classical clustering methods are not designed to find outliers in the data. These algorithms need to be modified to attain anomaly-detection capabilities. For example, a simple k-means clustering algorithm [42] does not account for outliers and performs clustering with the assumption that no outliers exist in the data. Therefore, k-means performs poorly in the presence of noise or outliers in the data, especially if these objects (noise or outliers) are located very distantly from normal data. In the process of capturing those distant outliers, k-means tends to group all other objects in the remaining required clusters. One of the ways to overcome this problem is to generate a large number of small clusters and then iteratively remove and merge those clusters until the desired number of clusters is reached [140, 141]. It is usually left to the user to determine when such removal and merging are affected by the presence of outliers. In other words, the question of the automatic identification of outliers, while at the same time automatically generating natural groupings that are not affected by the outliers is unresolved in clustering.

In first part of this chapter, the HAIS algorithm was proposed to automatically find clusters and outliers in the data simultaneously. However, the main focus of part I was on the clustering aspects of the algorithm alone. The aim of part II of this chapter is to extensively evaluate the HAIS for outlier detection on simulated and real-world datasets. The HAIS is based on the humoral immune response that secretes antibodies when triggered by the NIS. As far as we know, this is the first attempt to add outlier detection capabilities to nature-inspired clustering algorithms. The motivation here is to extend the HAIS algorithm to outlier detection and to show that it has the capability of finding clusters as well as outliers in the data simultaneously.

3.7 Outlier Detection – Literature Review

Outlier detection is an important area of research in data mining and is the process of detecting data objects which are significantly different from the rest of the data. Outliers can exist in the form of a single point or as a small cluster with significantly few objects associated with it. As mentioned earlier, outlier detection approaches can be classified into four main groups: distance-based, distribution-based, density-based, and cluster-based approaches [141, 142].

Distance-based approaches [146, 147] identify an object as an outlier if it is at least d_{min} distance away from g percent of objects in the data. The user has to specify g and d_{min}; these parameters could be difficult to determine *a priori*. The number of outliers detected by this method is highly dependent on d_{min} as different values of d_{min} can produce different numbers of outliers. Another distance-based approach presented in [148] is to rank each object with respect to its distance from g percent of objects in the data and the top k percent of objects in this list would be considered as outliers.

In the distribution-based approach a statistical model is developed based on the data and then an object is considered to be an outlier if it deviates too much from the underlying probability distribution [149-151]. One of the main drawbacks of both these approaches is that they are univariate in nature, whereas many real-world tests are multivariate. Prior knowledge of the underlying distribution is required, which makes these approaches inappropriate for most real-world data applications.

Distribution-based approaches consider the statistical distribution of attributes while ignoring the spatial relationship among objects. Density-based methods have been developed to find outliers in spatial data by considering attribute values and their spatial relationships together. In these approaches, the density of regions is calculated and objects in low density are considered to be outliers [152, 153]. A local outlier factor (LOF) [154] is used to identify objects as outliers relative to their local neighborhood densities. The LOF can find outliers more efficiently regardless of sparseness of data clusters, since it is based on the densities of a local neighborhood.

Despite previous research in outlier detection in data mining, the relationship of these four approaches to (unsupervised) clustering is still not fully understood. In cluster-

based approaches, small clusters are considered as outliers [140]. A two-phase outlier detection approach is presented in [155]. In the first phase, a modified k-means clustering algorithm is used to cluster data and then in the second phase a minimum spanning tree (MST) is constructed. The small clusters (trees with fewer nodes) are selected and considered as outliers. Almeida [156] proposed an outlier detection approach based on a hierarchical clustering algorithm. This method uses three steps. First, expected outliers and noise are removed from the data based on some density function, then clustering is performed using hierarchical methods with single linkage, and then finally (and optionally), the removed outliers and noise objects are allocated to clusters found earlier. Almost the same method is used by Loureiro [157], where cluster size is used as an indicator of outliers. The method uses hierarchical clustering and makes two basic assumptions: outliers are located far from normal clusters and their size is much smaller. Al-Zoubi [141] proposed a three-step methodology for outlier detection. In the first step the partitioning around mediods (PAM) algorithm [158] is used for data clustering and small clusters are considered to be outliers. In the second step, clusters obtained are fine-tuned further based on within-cluster distance measures. Finally, more remote outliers are removed from the existing clusters until a termination condition is met. A similar approach is reported in [159]. All current approaches to identifying outliers as part of a clustering process require a multi-stage approach. There has been no attempt, as far as we know, to integrate the identification of outliers with normal clustering in such a way that normal clustering is not affected but outliers are still identified.

3.8 Proposed Approach and Parameter Settings

The basic functionality of the HAIS algorithm used here is the same as previously explained in section 3.3, but a few parameters and most importantly the termination condition are changed. It was shown earlier (section 3.4.3) that in the presence of outliers the algorithm does not converge. The HAIS outlier algorithm makes use of this lack of convergence, or oscillation. We have fixed the termination condition of the algorithm at 15 cycles for all datasets. At the end of the first cycle 10% of memory cells were transferred to the next cycle and then in all cycles 10% more memory cells from the last cycle were carried forward. This is a form of incremental learning: as the number of cycles increases the number of memory cells transferred to the next cycle

(generation) also increases. Memory cells contain the information learned during the current cycle and transferring those memory cells represents the transferring of knowledge from the current generation to future generations. At the end of the algorithm, the memory cells can be used for validation or testing purposes. These memory cells also perform the role of data reduction by capturing the essential aspects of many samples in the form of a few exemplars.

The HAIS algorithm uses normalized squared Euclidean distance as a similarity measure, which can be biased towards higher-scaled data features. Therefore, all datasets used in the experiments are normalized (scaled between 0 and 1) to assign uniform weights to each feature. A constant uniform mutation rate of 5% is used for each feature. Negative clonal selection (the removal of antibodies that are too similar to each other) plays an important role in keeping down computational time while maintaining diversity in antibodies. A negative clonal selection threshold (NegT) of 0.0075 was used for all datasets; α was set to be variable depending on the dataset, whereas β and γ were set to 0.75 and 0.25 respectively.

In the original HAIS approach, at the end of every cycle a specific number of memory cells are transferred to the next cycle and all their antibodies deleted before starting a new cycle. In the case of incremental learning, the transfer of memory cells from one cycle to another increases with the number of cycles. This can lead to two variations: (a) no transfer of antibodies between cycles and, (b) before starting the next cycle antibodies are generated based on knowledge learnt in the previous cycles (memory cells). Memory cells depict a specialization of encountered instances and require higher affinity with antigens to trigger an immune response, whereas antibodies are generalizations of encountered data and can be activated upon relatively low affinity. In the context of outlier detection in data, if no antibodies are transferred between cycles (case ‘a’) the obtained outliers are identified based on affinity measure threshold (AT) of memory cells. On the other hand, if antibodies are also transferred between cycles (case ‘b’) the final outliers obtained are identified based on the AT of the B-cell antibodies (b-Abs).

3.9 Experimental Results and Discussion

Two simulated and four real-world datasets were used to test the outlier detection capabilities of the HAIS. The simulated data is used to test whether the algorithm can find clusters and outliers known to exist in the data. Three benchmark datasets, namely Iris, Breast Cancer Wisconsin, and Boston were used. While these datasets are relatively old, they are also well understood in terms of effects on clustering algorithms, allowing comparison between the outliers and clustering results obtained by the HAIS algorithm against other approaches from the literature. Finally, a new dataset consisting of medical doctor performance evaluations is used in an attempt to identify possible under-performing doctors.

3.9.1 Simulated Data 1

Simulated data 1 had three main clusters of varying sizes and shapes. There were a total of 197 instances with two features. The three main clusters constitute 178 instances while the remaining 19 instances are noise or outliers. The 2-D projection of this data can be seen in Figure 3.13. A mutation rate of 5.0% was set for all features and α was set at 20. The Ag to m-Abs stimulation level is set to be 10 times higher than Ag to b-Abs stimulation level. Various parameter values were tested and those mentioned above produced consistently better clustering results. The same rationale for parameter selection was used for the rest of the experiments in this thesis. The final clustering and outliers detected by the HAIS algorithm can be seen in Figure 3.13, where three major clusters are shown with different shapes and colors and outliers are identified by '+' and '*' signs. Solid marks within clusters are memory cells generated by the algorithm from the data that can be used for data summarization as well as determining membership of new and unseen Ags. A total of 57 memory cells were generated out of 178 instances by the algorithm, which represents a 68% data reduction. The DT parameter was set to 10. Due to the five outlier cases (in the upper right of Figure 3.13) forming a pseudo-cluster, three memory cells were also formed that summaries these five outliers.

At the end of the algorithm, a total of 16 clusters were found. Instance-wise cluster formation can be seen in Figure 3.14 (L), which clearly identifies the three main clusters and a relatively smaller fourth cluster that has only 5 instances (cluster-based outliers). All other clusters contain only one object (single point outliers).

The algorithm was run for 15 cycles and the total number of clusters obtained at the end of each cycle before removal of insignificant clusters (applying DT parameter) are shown in Figure 3.14 (R). It can be seen there that the algorithm found 12 clusters at the end of first cycle; that the lowest number of clusters were obtained in the fifth cycle (11 clusters); and finally, that the algorithm stopped at 16 clusters. This experiment indicates that the HAIS can clearly separate the outliers from the non-outlying cases (clusters/natural grouping) and can even cluster outliers if they are sufficiently similar to each other.

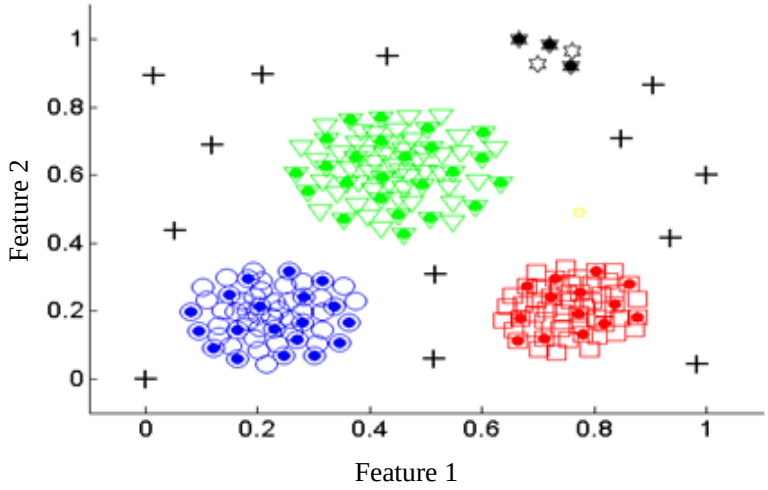


Figure 3.13: Final clustering obtained by the HAIS algorithm for simulated data 1, where three main clusters are shown using round, square and triangular shapes to represent the samples in the clusters.

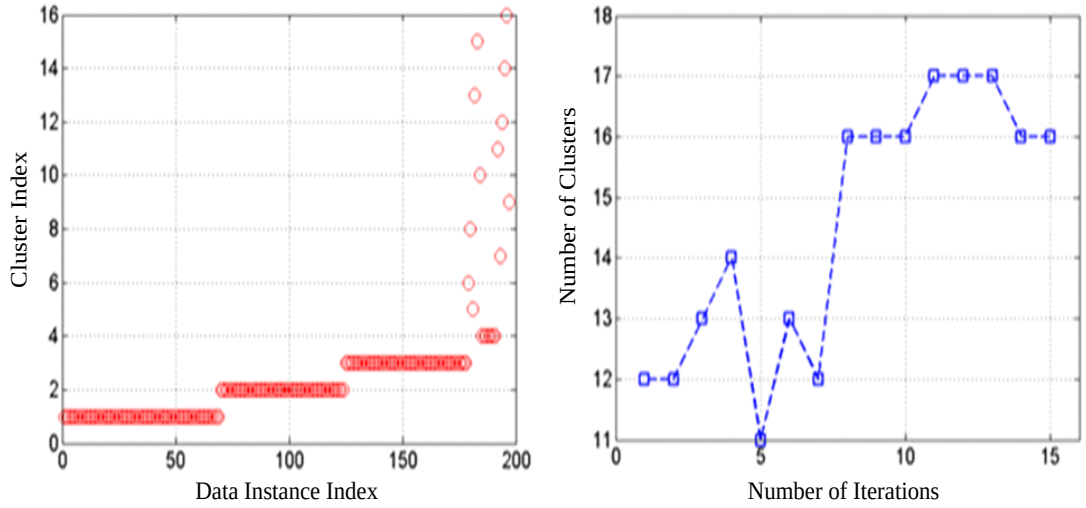


Figure 3.14: L: Instance-wised cluster formation of simulated data 1. R: Number of clusters obtained at the end of each cluster before applying the DT parameter.

3.9.2 Simulated Data 2

Simulated data 2 had two features and three distinct clusters, as shown in the Figure 3.15. The purpose of this experiment was to demonstrate the effectiveness of β and γ parameters in finding outliers in the data. Three experiments were run using the β and γ values of 0.75, 0.25; 0.5, 0.5; and 0.25, 0.75 respectively (Figure 3.16). The results shown in Figure 3.16 also demonstrate that the final number of memory cells obtained by the HAIS algorithm is dependent on the AT parameter. AT and number of final memory cells have an inverse relationship, such that if AT is high, the number of memory cells generated would be low and vice versa.

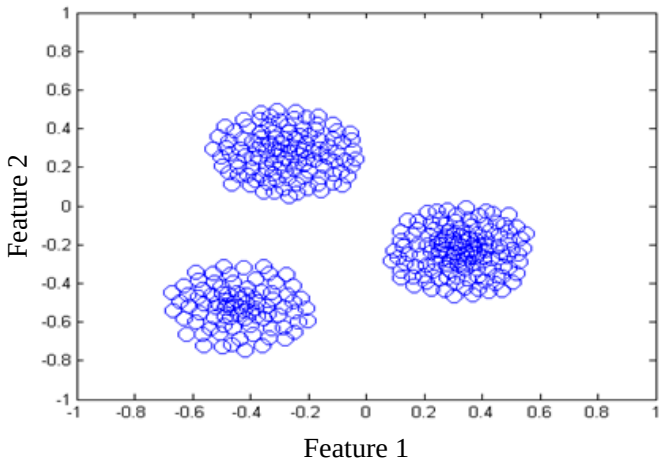


Figure 3.15: Original 2-D representation of simulated data 2, showing three distinct clusters

It can be seen in Figure 3.16 that when β and γ are 0.75 and 0.25 respectively (case A), the AT does not converge and keeps increasing with an increasing number of cycles. When β and γ are 0.5 and 0.5 respectively (case B) it takes more cycles to converge than when β and γ are 0.25 and 0.75 respectively (case C). Cases A, B and C in Figure 3.16 converge at final approximate average values of 0.25, 0.0425 and 0.025 respectively. If AT does not converge or converges at a higher value of AT, there is a chance that some of the outliers in the data will go unnoticed. The purpose of these experiments was to find outliers in the data, therefore β and γ are set to 0.25 and 0.75 respectively here. From this experiment, it can be concluded that β and γ play an important role in finding the final number of outliers in the data.

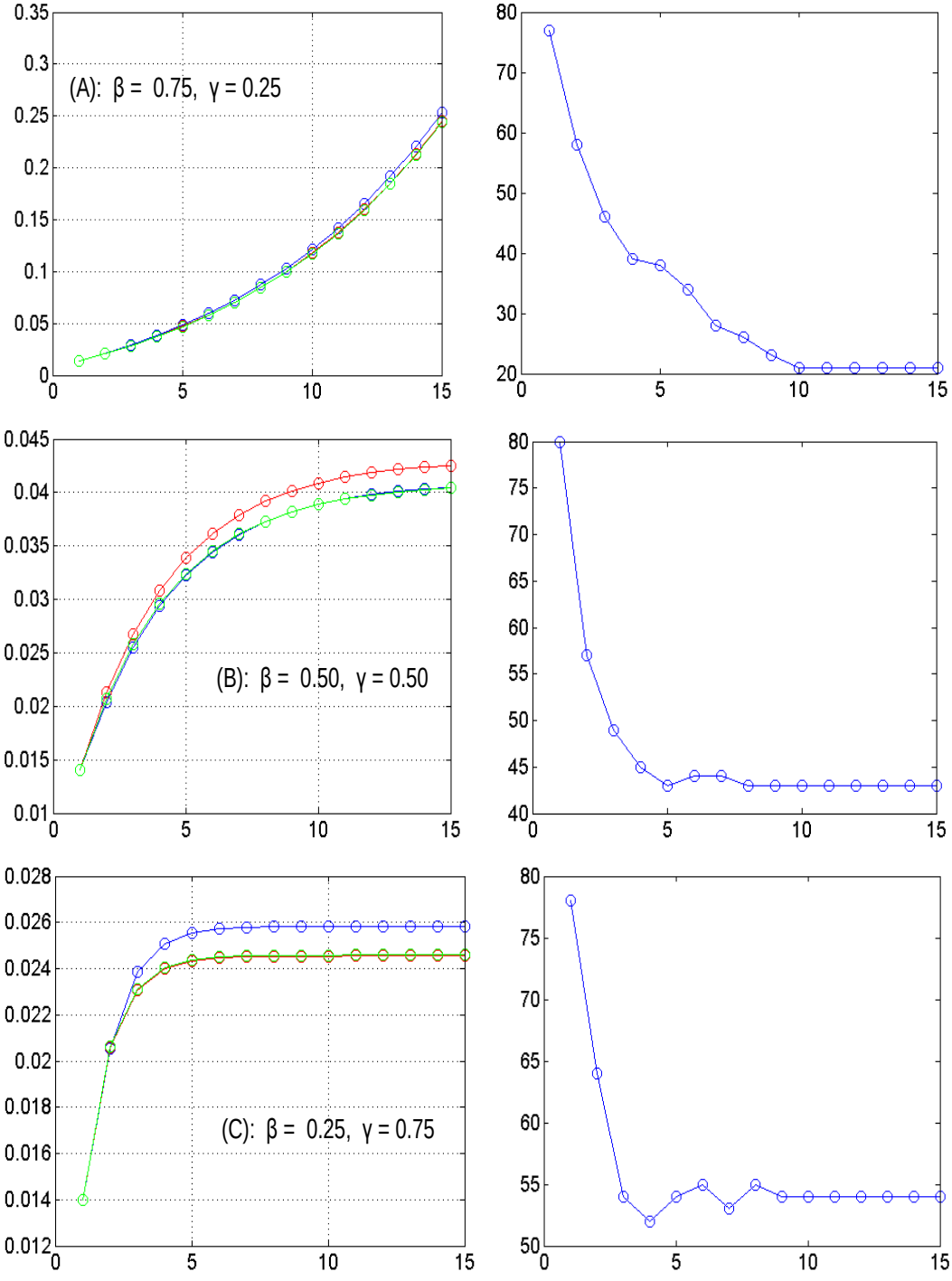


Figure 3.16: L: The behavior of various β and γ parameter values (the convergence of the AT parameter) is shown. R: The number of memory cells generated using the HAIS at the end of each cycle, is shown, for 15 cycles.

3.9.3 Iris Data

A mutation rate of 5% was set for all features and α was set to 15. The Ag to m-Abs stimulation level was set to be 10 times higher than the Ag to b-Abs stimulation level. The HAIS algorithm evolves as it gets exposed to Ags and pathogens. Therefore, the number of clusters as well as outliers is dependent on the order of Ag presentation. The distance similarity measure (AT) in the HAIS algorithm is dependent on the clusterings obtained. Therefore, the number and types of outliers detected are also dependent on the clusterings obtained. The results are shown in Table 3.4. The AT index represents the final affinity measure threshold obtained for each cluster, and the number of outliers for each cluster and total number of outliers obtained are also shown in the corresponding columns. Table 3.4 suggests that outlier identification is strongly dependent on the AT measure. The AT parameter has an inverse relationship with the total number of outliers: as the final AT value increases, the number of outliers decreases. The number of outliers for each AT value and in total are also provided in Table 3.4. The total number of outliers varied from 4 to 9 in the five runs using the same parameter settings. In the case of the Iris data, index (run) 4 was chosen to further explain the results, which indicate a total of 6 outliers in the Iris data.

Table 3.4: Five runs with the same parameter (α=15) for the Iris data

Index	AT-1	AT-2	AT-3	Outliers	Total
1	0.0167	0.0210	0.259	3, 1, 5	9
2	0.0167	0.0211	0.0273	3, 0, 3	6
3	0.0168	0.0215	0.0265	3, 0, 4	7
4	0.0167	0.0217	0.0268	3, 0, 3	6
5	0.0178	0.0222	0.0270	1, 0, 3	4

The results are explained using the confusion matrix. In a confusion matrix, each predicted class represents a column of the matrix, whereas each row represents instances of an actual class. The confusion matrix obtained for the Iris data using the HAIS clustering algorithm is shown in Table 3.5, which indicates 3 outliers each in class one and class three. The clusters obtained through the HAIS are labeled C1 to C6, whereas G1 to G3 represent the original class labels. It can be seen from Table 3.5 that the HAIS found three main clusters (C1, C2 and C3) containing 57, 47 and 40 instances,

respectively. In addition, it found three outlier clusters (C4, C5 and C6) containing 1, 3 and 2 instances, respectively.

Table 3.5: Confusion matrix for the Iris data

	C1	C 2	C 3	C 4	C 5	C 6
G1	0	47	0	1	0	2
G2	12	0	38	0	0	0
G3	45	0	2	0	3	0

Cluster centroids for all six clusters (including outliers) obtained by the HAIS algorithm are shown in Table 3.6, along with the average cluster centroids (Cluster Avg.). The Iris data has four features and the HAIS has found 6 clusters (3 true clusters and 3 outlier clusters). The mean value of each feature of each cluster, as well as cluster average, is provided in the last row of Table 3.6.

Table 3.6: Cluster Avg. for each feature for the Iris data

	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6
Feature 1	0.6187	0.1921	0.3825	0.0556	0.9167	0.3611
Feature 2	0.4032	0.5895	0.2619	0.1250	0.7222	0.9583
Feature 3	0.7291	0.0790	0.5215	0.0508	0.9153	0.0763
Feature 4	0.7453	0.0594	0.4798	0.0833	0.8889	0.0833
Cluster Avg.	0.6241	0.230	0.4114	0.0787	0.8608	0.3698

Table 3.7 shows the number of clusters obtained as well as the numbers of instances captured by each cluster, which indicates three main clusters and three outlier clusters. The averages of each cluster centroid and final AT measures for each cluster are also displayed in Table 3.7. Each cluster starts with the same AT value, but it is only updated if a cluster has been able to survive to the next cycle and not been killed by the DT parameter. It can be observed that all outlier clusters (clusters 4, 5 and 6) have the same AT value. This behavior arises due to the presence of the DT parameter, as by the end of each cycle all clusters below the DT value are deleted and the AT values are only updated if a cluster can survive. This indicates that all three outlier clusters are formulated in the current cycle. Previous work [159] using the PAM algorithm reported 8 outliers: 1 from class 1 and 4 and 3 from classes 2 and 3 respectively. The HAIS

algorithm found 6 outliers: 3 each from classes 1 and 3. The outliers (instance indices) detected by the HAIS algorithm in the Iris data are: 16, 34, 42, 110, 118, 132.

Table 3.7: Information regarding clusters and outliers obtained for the Iris data

Cluster	**1**	**2**	**3**	**4**	**5**	**6**
Instances	57	47	40	1	3	2
Cluster Avg.	0.6241	0.230	0.4114	0.0787	0.8608	0.3698
AT	0.0268	0.0167	0.0217	0.0171	0.0171	0.0171

Final clustering along with the outliers is shown in the 2-D projection in Figure 3.17, where different shapes and colors represent distinct clusters and outliers are shown by '+'. Solid round marks represent memory cells obtained by the algorithm. There are in total 81 memory cells, which indicates 46% data reduction.

This compression in the data in the form of memory cells is achieved when the Ag to m-Abs stimulation level was set to be 10 times higher than Ag to b-Abs stimulation level. Our experiments suggested that this data reduction percentage increases with the decrease in the Ag to m-Ab stimulation level. It is also dependent on the α parameter. The data reduction percentage increases with a decrease in α. Data reduction was 60% when α was set to 12 as compared with 46% for α=15.

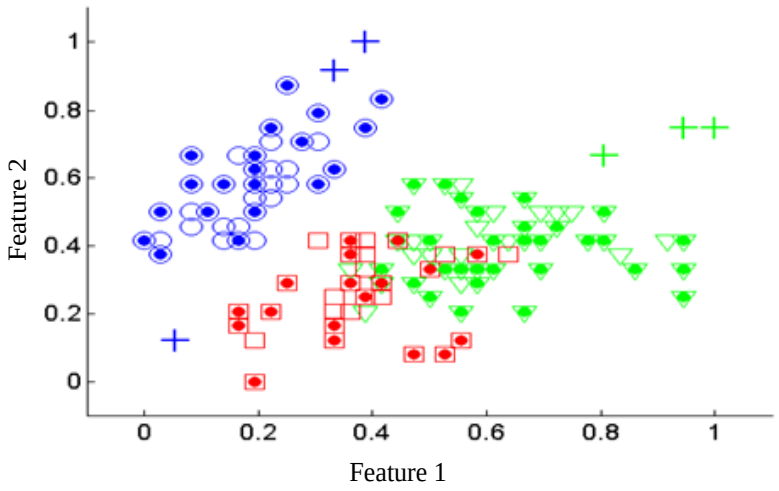


Figure 3.17: Final 2-D clustering obtained by the HAIS algorithm for Iris data, indicating 6 outliers identified by the '+' symbol.

3.9.4 Breast Cancer Wisconsin Data

A mutation rate of 5% was set for all features and α was set to 9. The Ag to m-Abs stimulation level was set to be 7 times higher than the Ag to b-Abs stimulation level. The algorithm generated 284 memory cells out of 569 instances obtained, which indicates 50% data reduction. The algorithm was run 5 times and the outliers detected in each run as well as the final AT parameter values obtained for the two main clusters can be seen in Table 3.8. The outliers' memberships to original cluster labels as well as the total numbers of outliers are also shown in Table 3.8. Index (run) 2 was selected to further explain the results for this dataset.

Table 3.9 shows the numbers of clusters obtained and also the number of instances captured by each cluster, which suggests two main clusters and seven outlier clusters. All outlier clusters are single point outliers except cluster 3 which has two instances. The average of each cluster centroid and final AT measure for each cluster are also shown in Table 3.9, which suggests that all outliers obtained are much closer to cluster 2 than cluster 1, even though 4 outliers originally belonged to cluster 1.

Table 3.8: Five runs with same parameter for the Breast Cancer Wisconsin data

Index	AT-1	AT-2	Outliers	Total
1	0.0391	0.0258	2, 4	6
2	0.0387	0.0266	4, 4	8
3	0.0374	0.0259	3, 8	11
4	0.0388	0.0262	3, 6	9
5	0.0384	0.0264	3, 4	7

Table 3.9: Information regarding clusters and outliers obtained from the Breast Cancer Wisconsin data)

Cluster	1	2	3	4	5	6	7	8	9
Instances	347	214	2	1	1	1	1	1	1
Cluster Avg.	0.1801	0.3280	0.3010	0.3387	0.4833	0.3605	0.4306	0.4696	0.5829
AT	0.0266	0.0368	0.0156	0.0156	0.0156	0.0156	0.0156	0.0156	0.0156

The confusion matrix obtained by the HAIS clustering on the main two clusters is shown in Table 3.10, which indicates 28 classification errors in the dataset. While there are no direct comparisons with previous outlier analysis on this specific dataset, Duan *et*

al. [140] report four single point outliers using a different version of this dataset (699 instances, 9 attributes). The average of 8 outliers per run using the HAIS is within an acceptable range for these datasets, given the differences in attribute numbers and instances.

Table 3.10: Confusion matrix for Breast Cancer Wisconsin data

	C1	C2
G1	197	11
G2	17	336

3.9.5 Boston Data

The algorithm was run on this dataset with a mutation rate of 5% and with α=5. The memory cell stimulation was set at 7 times higher than that for b-Abs. The algorithm was run 5 times, and outliers detected in each run as well as final AT parameter values obtained for the two main clusters can be seen in Table 3.11. The outlier clusters with numbers of instances as well as the total numbers of outliers are also shown in Table 3.11. The total number of outliers ranges from 14 to 21. The experiment with 15 outliers (index 1) was selected to explain the results.

Table 3.11: Five runs with the same parameter (α=5) for the Boston data

Index	AT-1	AT-2	Outliers	Total
1	0.0478	0.0963	1, 8, 2, 2, 2	15
2	0.0478	0.0948	5, 2, 8, 4, 2	21
3	0.0472	0.0946	8, 2, 1, 3	14
4	0.0488	0.0963	3, 5, 3, 2, 1	14
5	0.0472	0.0948	3, 8, 1, 2, 2, 2, 2	20

The HAIS algorithm found two main clusters with 118 and 373 instances in each (Table 3.12) and it also found 5 outlier clusters. Only one cluster is a single point outlier while four clusters have multiple outliers. Cluster averages are shown in Table 3.12 and indicate that outlier clusters 4, 5, 6 and 7 are located closer to the main cluster 1, whereas cluster outlier 3 is positioned closer to the main cluster 2. The algorithm generates 110 memory cells out of 491 instances (not considering outlier clusters), which indicates a 78% data reduction. Duan *et al.* [140] found 18 outliers in the Boston data whereas the HAIS algorithm found 15. Seven of the 15 outliers found by our

algorithm are also reported by Duan in [140]. The data indices obtained from the 15 outliers are as follows: 284, 357, 358, 358, 364, 365, 370, 371, 373, 413, 415, 369, 372, 411, 419.

Table 3.12: Information regarding clusters and outliers obtained from the Boston data

Cluster	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Instances	118	373	1	8	2	2	2
Cluster Avg.	0.498	0.344	0.401	0.601	0.482	0.530	0.473
AT	0.048	0.096	0.046	0.046	0.046	0.046	0.046

3.9.6 Doctor Questionnaire Data

The UK Chief Medical Officer's response to the fifth report of the Shipman inquiry [160] led to the UK Government endorsing periodic revalidation of all the UK's 200,000 doctors [161, 162]. Revalidation is seen as comprising the two strands of relicensing (confirming that doctors practice in accordance with the UK's General Medical Council's (GMC) generic standards) and recertification (confirming that doctors on the specialist/GP registers conform to standards appropriate for their specialty of medicine). The Peninsula College of Medicine & Dentistry located at the Universities of Exeter and Plymouth was commissioned to examine the potential utility of patient and colleague questionnaires in providing suitable evidence regarding the validation of doctors. This research has resulted in a number of publications [163-167]. There is an urgent need to consider another problem that is now starting to surface; how do we identify under-performing doctors who may warrant further scrutiny by the GMC, given the great reluctance of colleagues and patients to provide a negative rating due to the high stakes involved?

The final dataset used here is from a UK pilot study involving over 17,000 colleague responses (questionnaires) for 1050 doctors (average of 16 colleague responses per doctor). The questionnaire consists of 18 items (features) that ask colleagues to rate doctors on clinical ability, interpersonal skills and professional standards. Missing values (about 5% of all responses) are replaced by item means. Colleague responses were aggregated by doctor and the task is to identify doctors for possible further scrutiny and separate such doctors as outliers from those who do not require further scrutiny. Also, it is important to identify doctors whose performance does not warrant

placing them in the ‘further scrutiny’ category but nevertheless is a possible cause for concern (‘at risk’). Remedial measures, such as improved personal development on specific aspects of performance, may then be discussed with doctors. Since it can be expected that the vast majority of doctors perform well, the aim here is to cluster satisfactory doctors in a group, or in groups, that are separate from individual underperforming/at-risk doctors or groups of underperforming/at-risk doctors based on similarity and dissimilarity measures calculated from their scores on the 18 items. More information on the UK pilot study can be found in [165].

To compare the results of our HAIS clustering algorithm against a norm, a statistical measure for adequate performance was calculated as follows. First, the data was normalized (scale between 0 and 1), and the average of all the features was calculated for each doctor. Figure 3.18 shows the range of doctors’ data, where the y-axis shows the score of each doctor (average over all features). The satisfactory performance threshold is set at 0.5, meaning that all the doctors who score 0.5 or more are labeled ‘normative satisfactory’ (20 doctors, or approximately 2% of the cohort, fall below this threshold).

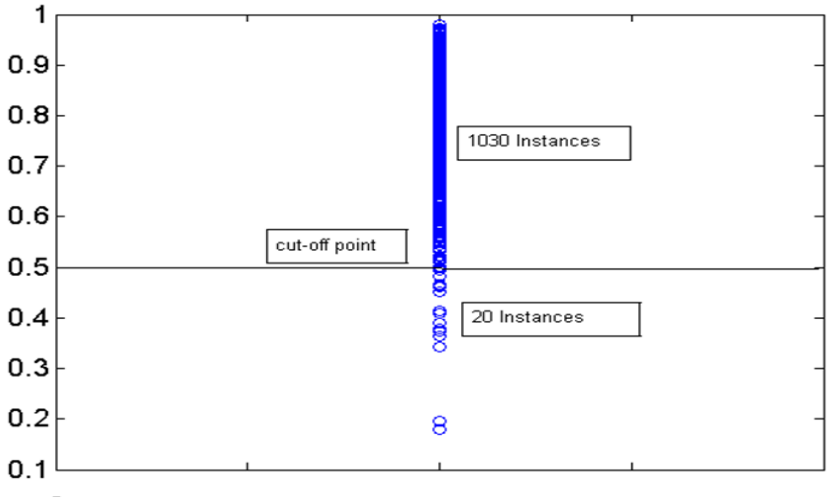


Figure 3.18: The average of all the feature values (mean score) of each doctor is represented by each blue circle (y-axis)

The task is to determine whether the HAIS is sensitive to two small but distinguishable groups (‘underperforming and worthy of further scrutiny’ and ‘at risk and worthy of further development’) in the context of the vast majority of doctors being satisfactory. The experiments were conducted with varying values of α parameter ranging from 8 to

15. The experimental result obtained using α=10 is shown in Figure 3.19. The HAIS algorithm found two clusters and six outlier clusters (single point outliers) in the data. It was also observed that when the value of 15 was used for the α parameter, the algorithm found 14 outlier clusters along with 2 normal clusters, which suggests that number of outliers detected is dependent on the value of α parameter selected.

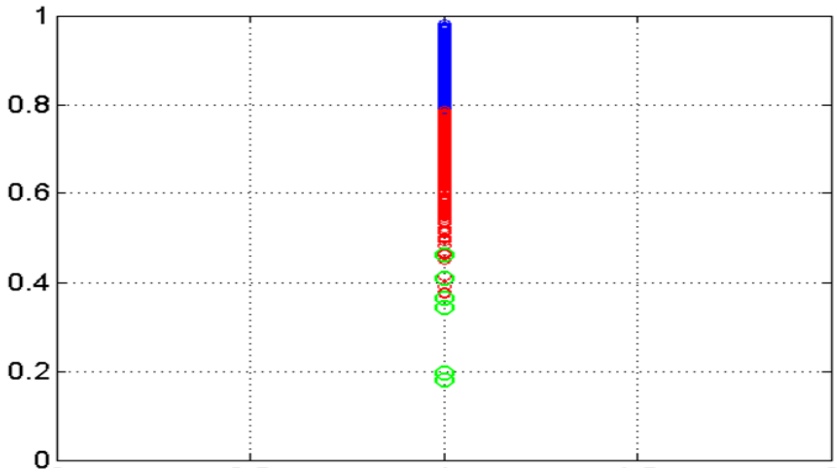


Figure 3.19: Mean score of each instance (y-axis) in the doctor dataset. The red and blue lines are clusters 1 and 2 respectively, whereas green circles indicate outliers found by the HAIS algorithm.

To explain the effectiveness of outlier detection (in this data), a two-step methodology was devised, where the HAIS algorithm was used to detect and eliminate outliers in Step 1 and a simple hierarchical clustering algorithm is used on the remaining data to find natural groupings in the data in Step 2. This approach was compared against standard hierarchical clustering on the complete doctors' data. In hierarchical cluster analysis (HCA), agglomerative clustering first assigns each case to its own cluster, followed by an iterative process whereby the two most similar clusters form a new cluster until one overall cluster results. Clusters that are added to each other can consist of single cases or multiple cases. The output is in the form of a taxonomy or hierarchical tree called a dendrogram.

When HCA is run on the complete dataset, 1032 (out of 1050) doctors are allocated to cluster one, and clusters two and three have only 15 and 3 doctors, respectively. This clustering behavior suggests that the data contains some outliers, because one cluster is assigned with more than 98% of the data. When these outliers are removed from the data using the HAIS algorithm and the HCA is performed on the remaining data (after removing 6 outliers), the three clusters obtained have 867, 165 and 12 instances

respectively. These results indicate three main groupings in the data: cluster 1 has 867 instances (satisfactory performance); cluster 2 has 167 doctors with possible cause of concern; and cluster 3 contains 12 under-performing doctors.

3.10 Summary

Experiments on simulated and real-world datasets suggest that the HAIS has the capability of detecting single point as well as cluster-based outliers. One advantage of the HAIS over other techniques is that it can find automatic clusters and outliers in the data simultaneously. The HAIS is an iterative algorithm, where clustering is performed first and then those clusters are evaluated for significance with the small-sized clusters removed. In the next cycle, those removed objects are again considered in the clustering step. Small-sized clusters are removed at the end of each cycle but outliers stay in the system until the end of the algorithm. The experimental results show that the proposed approach gives effective results when applied to different real-world datasets. They also demonstrate that removing outliers from the data can help find better clustering outcomes (e.g. in doctor questionnaire data).

Non-convergence is represented by the oscillation of the HAIS as the DT and NT parameters reduce the number of clusters at the end of each cycle (Figure 3.5 [R]). That is, during one cycle of the HAIS, 8 clusters were found. These 8 clusters consisted of the three main clusters (Figure 3.5 [L]), one cluster consisting of 7 samples, and four outlier clusters of one sample each. At the end of each cycle, DT and NT reduce the number of clusters and the process of finding 8 clusters is repeated during the next cycle. This is in contrast to normal HAIS behavior, which converges on a stable number of clusters after a number of iterations. In other words, lack of convergence is a possible indicator of the presence of outliers in the data. It is possible that further analysis, such as using a specially adapted discrete Fourier transform, could reveal components of different frequency (e.g. clusters of different density) that could help to identify outliers. Oscillation, or lack of convergence, is not considered a desirable feature in machine learning, but for the identification of outliers in the data it could well be.

In this chapter we have presented a novel unsupervised clustering algorithm inspired by the human NIS. Some of the salient features of the HAIS algorithm are as follows:

1. Multi-Layered Approach: The proposed HAIS algorithm works on three distinct layers: (1) the memory cells layer, (2) the antibodies layer, and (3) the B-cell layer, in that order. The first two layers try to capture a new antigen, but if both layers cannot capture the invader, a new B-cell is generated to mount an appropriate response.

2. Discriminates Self from Non-Self at Learning Stage: Unlike CLONALG and aiNet, the HAIS explicitly labels antigens at the learning stage by allocating them to respective B-cells. In the absence of assigning labels to antigens, an unsupervised algorithm is only useful for screening the data. But this class association can be changed within the same or different cycles.

3. More Active Role of Memory Cells: Previous approaches only expose antibodies to antigens and memory cells remain hidden from antigens. However, the HAIS uses both antibodies and memory cells to capture antigens. This feature is important since in the NIS memory cells play a more active role and help the immune system to capture antigens. Memory cells are more specialized cells and therefore they require more stimulation than antibodies to be activated.

4. Outlier Detection Capabilities: While devising a novel cluster analysis technique, researchers usually completely ignore the role and importance of outliers. The HAIS algorithm has the capability of performing cluster analysis as well as outlier detection. The presence of outliers does not affect the performance of the HAIS clustering results.

5. Specific Reference to Plasma Cells: The HAIS algorithm makes specific reference to plasma cells in the generation of antibodies and memory cells.

6. A Step towards Online Learning Algorithm: A key characteristic of the NIS that appeals to machine learning researchers is its ability to perform life-long and online learning. We do not claim that the HAIS in its current state is an online learning algorithm. However, some characteristics such as mergence and removal of clusters certainly suggest that the HAIS can be modified into an effective online learning algorithm.

7. Evolving Affinity Threshold (AT) Parameter: Most of the algorithms in the literature use an iterative approach, but keep the affinity threshold measure (AT) constant, meaning an extra step is required to perform extensive statistical methods to select a reasonably good AT value. By contrast, the HAIS updates its AT value at the

end of each cycle. Also, it was shown in the experimental section that convergence of AT can be achieved through the β and γ parameters.

8. Role of Negative Selection Threshold (NST): NST is used to control antibody populations. This concept was originally used in negative selection algorithms (see section 2.3.1 for details) for generating non-self receptors from self-receptors. In the HAIS, negative selection is performed on antibodies of the same cluster (group) to increase natural selection pressure on other clusters. It is also important to only perform intra-cluster negative selection to accommodate overlapping clusters.

9. Incremental Learning: Knowledge learnt in previous generations must be transferred to the following generations. The HAIS algorithm uses a form of incremental learning where a percentage of the memory cells formed in a previous cycle is carried forward to the next cycle, and this percentage increases with the number of cycles. This transfer of memory cells provides momentum and direction to clustering.

In this chapter, a novel HAIS algorithm inspired by the concepts of the adaptive immune system has been proposed. It was demonstrated via experimental results on simulated and benchmarked real-world datasets that effective clustering and outlier detection capabilities can be obtained simultaneously by remaining faithful to immune system metaphors. Section 3.4 (experimental results) showed that the HAIS is a stochastic algorithm and its average performance is no worse than the standard statistical methods such as k-means and hierarchical clustering algorithms. The stochastic nature of the HAIS algorithm is mainly due to its random selection of a 10% data sample as a starting point and random order of antigen presentation in each cycle. In the HAIS, each random order of presentation can potentially produce different clustering solutions, even by keeping all remaining parameters constant. In the next chapter, it will be shown that how different models (clustering solutions) could be built from the random order of antigen presentation, and how we can select and evolve different local models to construct a global model in order to improve clustering results. Another core component of the HAIS algorithm is mutation, which has not been extensively studied in this chapter. Later chapters will look closely into mutation parameter, which helps immunoglobulin receptors to proliferate, and differentiate to capture similar antigens, as well as be responsible for affinity maturation in the HAIS algorithm.

Chapter 4

Population-Based HAIS Clustering Algorithm

In the previous chapter, we proposed a novel artificial immune system (AIS) algorithm called the HAIS, which was inspired by humoral-mediated immunity and uses hypermutation to simulate the way natural immune systems (NISs) refine their B-cells and antibody receptors in response to pathogens. The HAIS algorithm is a stochastic algorithm (see section 3.4 and Figure 3.8), meaning it produces variable clustering results in different runs. The outputs (final clustering results) are highly influenced by the order of antigen presentation. The HAIS, like other AIS algorithms, is a decentralized process where antigens are trapped by antibodies based on local affinity information. Antigens are picked up from the antigen pool purely at random. In addition, B-cells generate a new population of antibodies based on already seen antigens, to trap future antigens. It has been suggested in chapter 3 that the final structure of B-cells (clusters) greatly depends on the order of antigen presentation, given a certain affinity threshold (similarity measure) criterion. The main motivation of this chapter is to explore this characteristic (various orders of antigen presentation) of the

HAIS algorithm using a population-based approach in the problem domain of unsupervised learning. The proposed population-based approach can be regarded as a metapopulation, as here we are incorporating a micro-level process (an AIS) with a macro-level population-based approach. At the micro-level, a population consists of antibodies, memory cells and B-cells; at the macro-level, different clustering solutions constitute the population which are generated using different antigen presentation orders. The rest of the chapter is structured as follows.

The rationale and basis for adopting a population-based approach is discussed in section 4.1. A descriptive two-step population-based approach and its algorithmic explanation is then presented in section 4.2, where the number of clusters is obtained in Step 1 using the HAIS and then in Step 2 an evolutionary (population-based) approach is used to further enhance the cluster quality. Convergence in the fitness of populations of individuals (clustering solutions) is achieved through transferring memory cells from one generation to another, as explained in chapter 3. Section 4.3 demonstrates the feasibility of the proposed approach by performing experiments on benchmarked real-world datasets. Additional results show the effectiveness of the crossover operator at the population level. The summary of this chapter is provided in section 4.5.

4.1 Introduction

The main purpose of cluster analysis is to discover patterns in data samples and separate those samples into groups in such a way that each group has maximum within-group similarity and, ideally, maximum between-group dissimilarity. Clustering is a form of 'unsupervised learning' in that metrics for estimating similarity and dissimilarity are not dependent on an objective top-level measure of which group or cluster a sample should belong to. Nature-inspired computing researchers have noted the similarities between unsupervised learning and decentralized, bottom-up processes in nature, with many nature-inspired clustering algorithms being proposed over the years based on genetic algorithms (GAs), ant colony optimization, and particle swarm optimization [21-24]. Such approaches typically derive their inspirations from macro-level biological phenomena, i.e. at the level of individual organisms (ants, particles) or populations of individuals (GAs). As our understanding of nature has grown, researchers have started to focus their attention on much deeper natural processes for designing and developing new computational algorithms. One such example is the NIS, where micro-level

interactions between antibodies and antigens lead to the macro-level property of an organism remaining alive by distinguishing between pathogens (which must be reacted to) and self (which must not be reacted to), with no macro-level or micro-level executive control, as far as we are aware. Some of the principles of an immune system, such as mutation and fitness thresholds, are also shared by other biologically inspired computational methods such as GAs, genetic programming and evolutionary computing in general.

The HAIS algorithm proposed in chapter 3 is inspired by the humoral-mediated response that is triggered by the adaptive immune system. It is a stochastic algorithm, so different runs of this algorithm using the same parameters can produce different clustering results. Previously reported results (see chapter 3, Figure 3.8) of the HAIS algorithm indicate that its average performance in terms of clustering solutions found is no worse than classical k-means and hierarchical clustering algorithms. Clustering results obtained by the HAIS are largely dependent on the order of data presentation. Given a dataset of n objects, there are $n!$ ways the data can be presented to the HAIS. This stochastic indeterminacy is typically addressed in two ways. The first way is to fix the pathogen presentation order, which means that the final model is acknowledged to be just one possible model out of all the models that can exist. A second way is to accept different presentation orders of data and construct a number of models that, by and large, return the same result. Rigorous statistical measures can also be used in inductive arguments to show that the result is, to as high a degree of probability as one wishes, the best. A feature of both methods is that no attempt is usually made to merge the many models that result into a 'super-model' which preserves the best aspects of the various individual models. Also, no attempt is made to separate objects that are not a problem for analysis from objects that are a problem for analysis. In other words, each model that is constructed typically does not 'learn' or evolve from the previous models that were constructed.

Hart and Timmis [10] proposed that an immune system must be embodied, since natural systems do not work in isolation, and made suggestions for integrating AIS with other natural-inspired approaches such as neural networks, swarm algorithms and GAs to find out the true potential of immune system-based algorithms. However, the way that such integration should occur, taking into account the problem domain, was left open. In the experiments to be described below, we attempt to improve the performance of the HAIS

algorithm by integrating micro-level processes of cluster formation with macro-level processes of allocation of samples to clusters, where the micro-level consists of initial humoral-mediated cluster formation and the macro-level of GAs that optimize allocation of samples to clusters. To achieve this integration of micro- and macro-level processes, a two-step algorithm is proposed in this chapter. In Step 1 initial data partitioning and the appropriate numbers of clusters are obtained using the the HAIS algorithm, and in Step 2 those obtained groupings are optimized in terms of data allocation using an evolutionary approach. The convergence in population of solutions is obtained by the incremental transfer of memory cells (knowledge acquired through interaction with pathogens) from one generation to another.

The idea of hybridizing AIS algorithms with evolutionary approaches is not new [168-170]. Potter *et al.* used the co-evolutionary approach to evolve a population of antibodies for concept learning [168]. Ahmadi and Maleki [169] used a similar approach to evolve AIS for network security applications. In separate research, Louis and McDonnell [171] incorporated a GA with case-based memory (of past knowledge) to obtain better performance and fast convergence on problems such as combinational circuit design, asset allocation and job shop scheduling. This approach is called CIGAR that periodically injects previously solved problems into a GA's population (random population). In our proposed population-based HAIS approach, we integrate concepts of [168, 169] with [171] in a novel way, where memory cells are used for reinforcement learning and a GA for adaptability. The aim is to combine the advantages of both paradigms so that (a) the HAIS acquires knowledge through interaction with antigens, which are stored in the form of memory cells for faster convergence; and (b) the GA provides enhanced search capability and adaptation [171].

In the population-based HAIS approach, the evolved immune system of each individual (here called a clustering solution) is tested against evaluation criteria (environmental factor/s) and only the fittest and best individuals (solutions) are carried forward to the next generation, while all remaining individuals are discarded. Reinforcement learning is performed at both micro and macro levels. Elitism is adopted as the selection strategy: copying the best individuals into the next generation for the purpose of preserving the best solutions obtained so far ('survival of the fittest'). Elitism ensures that the fitness of the next population can never reduce from the population of the previous generation. This phenomenon helps in the rapid convergence of the population.

An external clustering validation criterion is used to evaluate the fitness of each individual in the population. Our aim in this chapter is not to compare the results obtained using the population-based HAIS approach with existing state-of-the-art clustering techniques. Instead, we attempt to establish that micro-biological nature-inspired models such as HAIS can be incorporated with macro-biological processes at the population level to develop novel nature-inspired clustering algorithms, and also to demonstrate that the performance of the HAIS algorithm can be improved using a population-based approach.

4.2 HAIS Algorithm: An Overview

In this section, a brief summary of the HAIS algorithm is presented. A full explanation of this algorithm can be found in chapter 3. The main components of the HAIS are B-cells, antigens (Ags), memory cells, antibodies (Abs), and the affinity threshold (AT). The algorithm starts with 10% of data instances and considers each instance as an individual B-cell. B-cells are clusters for storing data and these B-cells generate memory cells (a 'synopsis' of data so far captured) as well as Abs. An Ab is an array of values, one for each attribute present in the data. Samples are captured or trapped by the Ab that is closest in value to their attributes (subject to a threshold of proximity). Two types of Abs are produced: one is generated by B-cells (b-Abs) while others are generated by memory cells (m-Abs). The similarity between an Ag and Abs is calculated by the pair-wise square normalized Euclidean distance.

The HAIS algorithm works mainly on three layers, as shown in Figure 4.1. At the first layer, m-Abs try to capture Ags. If the capture is successful (subject to satisfying the proximity threshold), the stimulated m-Ab brings the Ag to its respective B-cell. If not, the Ag goes to the second layer, where it is compared against existing b-Abs. If an Ag is captured at this level, the stimulated b-Ab will bring the Ag to its respective B-cell, which holds on to it until the end of the cycle. Moreover, after capture the B-cell produces a memory cell that further generates an m-Ab which is an exact copy of the captured Ag. B-cells also produce mutated b-Abs, which are non-identical copies of the captured Ag to allow the B-cell to capture other Ags that have similar but not identical attribute values. If even this layer fails to capture the Ag, a new B-cell is generated just for this particular Ag which will generate b-Abs. After each successful capture of an Ag, similarities between B-cells are calculated using a between-cluster metric and, if the

similarity between two B-cells is greater than the network measure threshold (NT), both B-cells form an inter-connected cluster (cluster mergence).

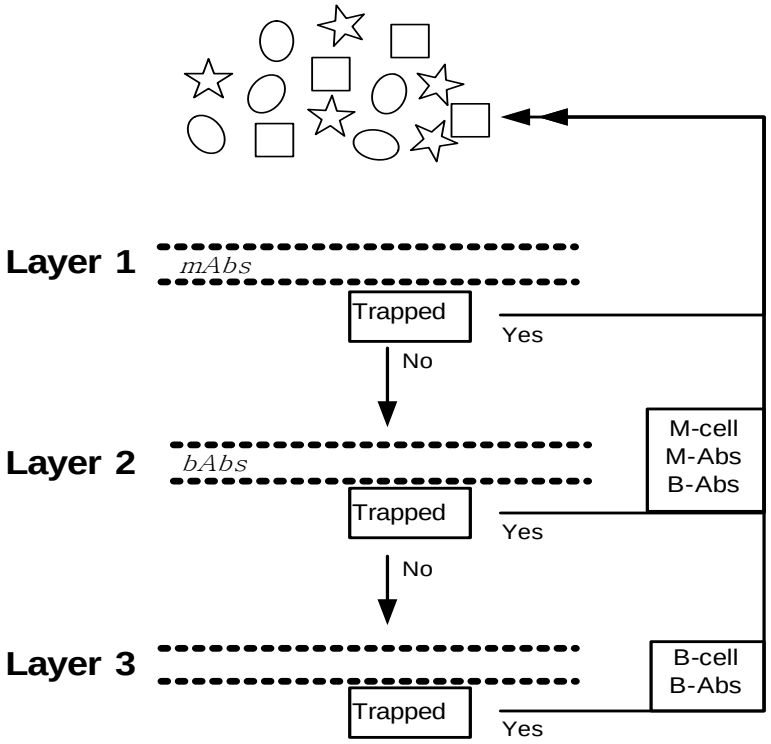


Figure 4.1: A snapshot of the HAIS algorithm proposed in chapter 3

This is an iterative algorithm: at the end of each iteration, B-cells are evaluated and less stimulated B-cells (small clusters) are removed and their Ags are released, based on a threshold criterion called DT. Surviving B-cells update their centroids according to the samples captured. All the Abs generated by B-cells or memory cells are also released, whereas M percent of memory cells near to the centroid of B-cells are carried to the next cycle. All the parameters – AT, NT and DT – are updated before the start of the next cycle. This whole process is repeated until there is no change in the number of surviving B-cells for two consecutive cycles. The AT parameter is used to capture similar Ags whereas the NT parameter is used to merge B-cells that are close to each other. The algorithm starts with the same value for AT and NT. The NT parameter decreases whereas AT increases with iterations. At the end of each cycle AT and NT are updated. More details about these parameters can be found in chapter 3, section 3.3.

4.3 Population-Based HAIS Algorithm

A descriptive overview of the proposed algorithm is provided below; a full explanation follows the algorithm.

1. Initial population of clustering solutions is obtained from HAIS (initial populations of clusters and their antibodies)
2. Generate new population (of clustering solutions) by exposing individual clustering solutions to random order of pathogens
3. Next population := existing population + new population
4. Generate off-spring population (of clustering solutions) by performing crossover on Next population
5. Total population := Step 3 + Step 4 (Next population + off-spring population)
6. Evaluate and select 'H' clustering solutions
7. Go back to Step 2 until termination condition

4.3.1 Initial Population

The original HAIS algorithm is run *k* times and out of those *H* clustering solutions are selected as the initial population. Once *H* clustering solutions are selected, just as in any standard HAIS algorithm cycle, all Abs are killed, the B-cells readjust to the new centroid position, and 10% of the memory cells closest to centroid of B-cells are kept while all remaining memory cells are discarded. New Abs are generated from surviving memory cells. The initial population of clustering solutions is then exposed to all Ags (using different order of presentation) one at a time (at random), to generate new population of clustering solutions (Step 2).

4.3.2 Generating New Populations

The main difference between the standard HAIS algorithm and the population-based HAIS algorithm described above is that mutation of Abs by B-cells after the successful capture of an Ag is now complemented by a crossover (see section 4.3.3) involving the population of clustering solutions formed at the first step. The subsequent steps (Steps 2-6) constitute optimization of the initial clustering solutions. After Step 1 the number of clusters stays constant (although clustering solutions can evolve through mutation of their Abs) and only the membership of samples (Ags) changes. Layer three of the original HAIS algorithm is now revised to take into account the two-phase approach (see Figure 4.2) which is a one-shot approach. If an Ag within a clustering solution cannot be trapped at the first two levels, instead of generating a new B-cell (cluster) the Ag is put back into the pool for re-selection later. This revision to the original HAIS

algorithm ensures that the number of clusters is kept constant within each solution after the initial decision on number of clusters at Step 1.

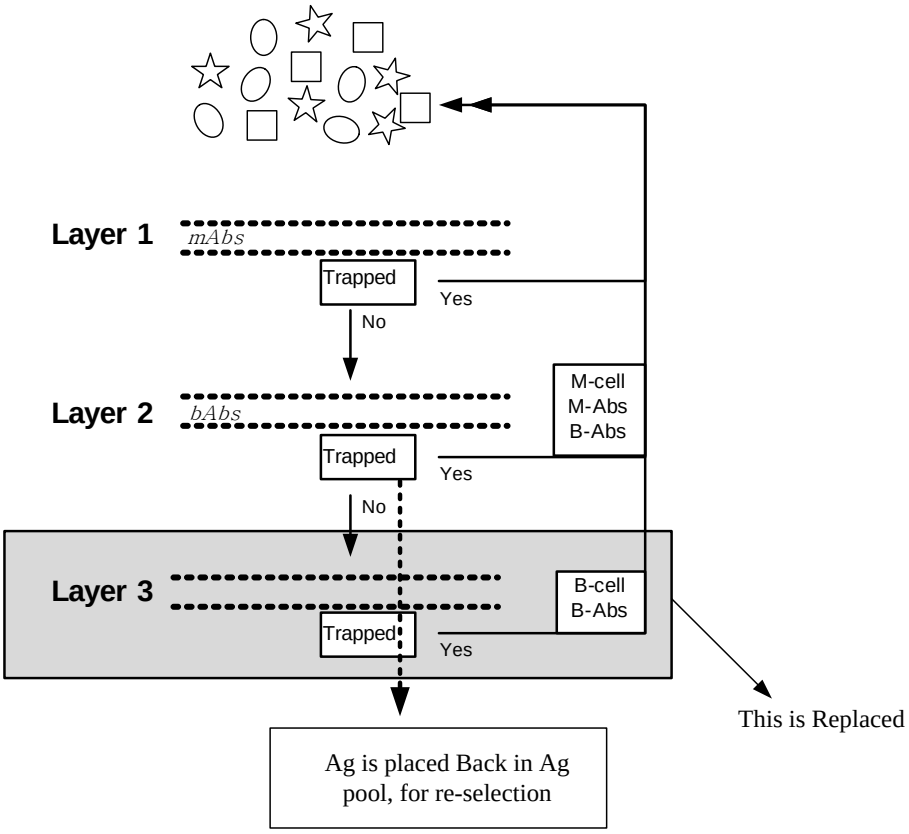


Figure 4.2: An overview of the revised one-shot HAIS algorithm and revised layer 3, for keeping population of B-cells constant

In chapter 3, in the standard HAIS algorithm, an Ag was randomly selected and exposed to the population of memory cells as well as the population of Abs to be captured and brought to the appropriate B-cell. Once an Ag is exposed, two scenarios can arise: (1) the new Ag is captured and assigned to the appropriate B-cell; or (2) if the new Ag is not captured; a new B-cell is generated to mount an appropriate immune response to the Ag. However, in this chapter, we have implemented the HAIS algorithm for a fixed (pre-defined) number of clusters. This means that there is no need to generate new B-cells. An Ag is selected at random and presented to the population of memory cells and Abs. If an Ag is not trapped by memory cells or Abs, it is put back into the Ag pool for re-selection later. This process is repeated until all Ags are captured by B-cells.

At the start of each generation during the second phase of the algorithm, a new population of clustering solutions is generated through a random order of pathogen

exposure (Step 2) and crossover (Step 4). The population at the start of new generation carries B-cells position (centroids) and selected memory cells from the last generation. At the end of each generation all Ags are set free for fresh random selection.

4.3.3 Crossover Operator

Two clustering solutions were randomly chosen as parents for crossover. In this HAIS algorithm, B-cells represent data clusters; therefore, a B-cell at random was picked and swapped between the two selected parents to generate two new offspring clustering solutions through a single-point crossover (see Figure 4.3). At the point of crossover, it is made sure that the same B-cells (the same cluster index) are swapped between parents to generate valid offspring. For example, assume there are two parents, each having 'safe' and 'danger' class labels. While performing crossover, a constraint must be applied so that swapping can only take place between the same classes (i.e. one can only swap a 'safe' of one parent with a 'safe' of another parent and vice versa). A complete transaction of generating offspring from two randomly selected parents can be seen in Figure 4.3. The single-point crossover is sufficient for the experiments performed in this chapter, as one (whole) cluster is exchanged between two clustering solutions and the datasets used in this chapter do not have more than three classes/clusters.

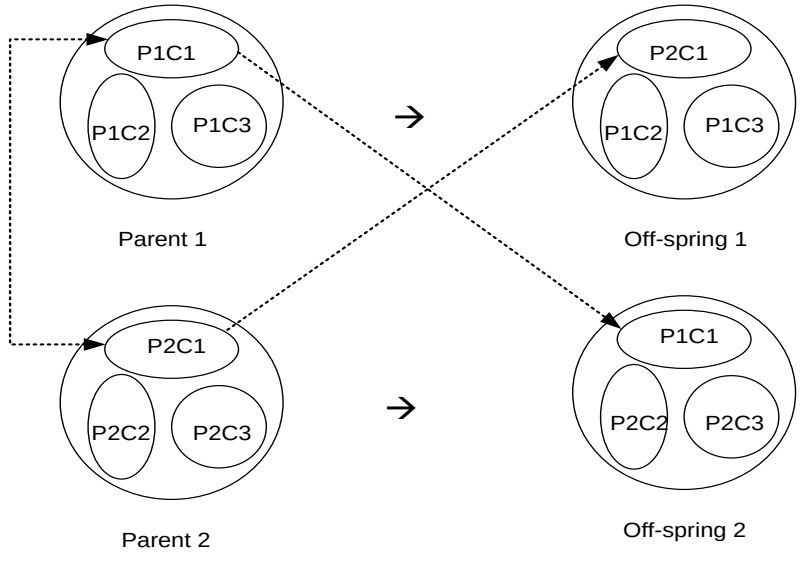


Figure 4.3: Single-point crossover between two parents producing two offspring. Parent 1 and Parent 2 (P1 and P2) are two randomly chosen clustering solutions, consisting of three clusters (C1-C3). Shown here is P2's C1 swapped with P1's C1 to produce two new clustering solutions as offspring. All Ags, memory cells and Abs attached to a cluster involved in crossover are also transferred to the offspring clustering solution.

4.3.4 Fitness Function

Clustering solutions obtained in each generation were evaluated using a cluster validity criterion. Several cluster validity measures have been proposed in the literature [172-174]. Cluster validity techniques can be classified into three groups: external criteria, internal criteria, and relative criteria [173]. External criteria are used to compare the clustering results with pre-defined partitions or class labels. The Rand index, Jaccard coefficient and Fowlkes-Mallows index [1] are examples of external cluster validity criteria. Internal criteria only consider the information within data to produce a quantitative measure about obtained clusters. The Davies-Bouldin index, Dun index, silhouette index and SD validity index [172, 175] are examples of internal cluster validity criteria. With relative criteria, clustering solutions obtained by the same algorithm using different parameters are compared with each other to find optimal clustering. Here, the number of clustering errors found by each individual clustering solution against true class labels are used to calculate fitness of each individual (external criteria). Therefore, the objective function used here minimizes clustering errors. All the datasets used in this chapter have class information along with original data, which makes it possible to use this criterion.

4.3.5 Population Evaluation

The selection strategy that follows crossover is rank selection. The population of clustering solutions at any stage consists of (1) the best clustering solutions found in previous generation; (2) the population of clustering solutions generated by the exposure of a random order of Ags during the current generation; and (3) offspring clustering solutions generated by crossover (from current generation). The H best solutions are selected for the next generation from the existing population of solutions. Here, our population-based approach adopts elitism to preserve the best clustering solutions generated. Therefore, at the selection stage, the best clustering solutions from the previous generation are also considered for re-selection.

4.3.6 Termination Condition

The termination condition is user-defined and, in the experiments below, is fixed at 12 generations. As explained earlier, incremental learning through the transferring of memory cells is performed as the number of generation increases. The algorithm starts

with an initial 10% transfer of memory cells with a 10% increment in subsequent generations, which means there would be 100% transfer of memory cells at the end of 10th generation. After that point there would be no more change in cluster membership of Ags. Therefore, the termination condition was set at 12 generations.

4.3.7 Reinforcement Learning through Memory Cells

Nature-inspired algorithms are characterized by reinforcement learning. For example, GAs select chromosomes for the next generation from the whole population of parents and offspring to provide the best opportunity for evolving globally best solutions. Ants in ant colony optimization perform reinforcement by attaching a pheromone to the travelled path; the more frequently used paths therefore have more pheromone to attract more ants. Particle swarm optimization algorithms achieve reinforcement learning by directing their movements through the local and global best solutions found so far. The standard HAIS approach performs reinforcement learning through transfer of memory cells from one generation to the next. Here, an incremental reinforcement learning technique is used, where the number of memory cells transferred to the next generation increases as the number of generations increases. In the experiments below, 10% incremental transfer of memory cells is used, which means that in the first generation 10% of memory cells are transferred to next generation and in second generation 20% are and so on (see Table 4.1). This is a form of transfer of knowledge from a previous generation to the next generation. Once the transfer of memory cells reaches 100%, there will be no further change in Ag allocation to B-cells, which helps in achieving convergence. The last two generations in Figure 4.4, for example, show that the algorithm finds local optimum solutions with no changes in value of fitness function. One advantage of such an approach is that not only the population of best selected individual converges but also the whole population of solutions converges as well. Here, before the start of generating a new population of clustering solutions, new Abs are generated based on existing memory cells in the system (transferred from the previous generation). These steps are important to provide convergence and direction to clustering solutions.

4.4 Experimental Results

The experiments were conducted on datasets with varying sizes and clusters. Five real-world datasets namely Iris, Wine, Thyroid, Breast Cancer Diagnostic and Breast Cancer Wisconsin Original (see appendix A) were used to show the effectiveness of the proposed algorithm. All these datasets are normalized and scaled between 0 and 1. The HAIS algorithm is dependent on three parameters: α, β and γ. The parameter α is a scalar value which controls the tightness of boundaries among the clusters whereas β and γ helps the AT parameter to converge. β and γ are set at 0.25 and 0.75, respectively, in the experiments below. A mutation rate for evolving Abs of 5.0% is used for the Iris, Wine and Breast Cancer Diagnostic datasets, and a 10.0% mutation rate is used for Thyroid and Breast Cancer Wisconsin Original datasets. The termination condition was set at 12 generations, since this was found sufficient to lead to convergence across all the datasets. The initial population was set at 10 (i.e. the number of runs of the standard HAIS algorithm). The population size (clustering solutions) was set at 100 with subsequently another 100 offspring generated through crossover. At the end of each generation, the 10 best clustering solutions were selected and carried to the next generation, as can be seen below.

1. Initial population = 10
2. New population (Population size) = 100
3. Next population = 110 (sum of Step 1 (or 6) and 2)
4. Offspring population size = 100
5. Total population = 210 (sum of Steps 3 and 4)
6. Best population selected = 10

Table 4.1: The map of memory cell transfer in each generation

Generations	1	2	3	4	5	6	7	8	9	10	11	12
M-cells transfer (%)	10	20	30	40	50	60	70	80	90	100	100	100

The Iris data has three classes consisting of 50 instances each. The dataset has four features (Sepal Length, Sepal Width, Petal Length and Petal Width) and three classes (Setosa, Versicolor and Virginica). The standard HAIS algorithm was run 10 times on the Iris data to generate the initial population. This initial population had an average

error (average fitness of initial population) of 21.90, as can be seen in Figure 4.4 (L) (along the x-axis at zero index) and indexes 1 to 12 represent the 12 generations. The average error among the best selected individuals at the end of first generation decreases to 6.4, with more gradual decrease in error subsequently. The error curve decreases until the 9th generation and then stabilizes at 3. The same trend can be observed in the average error curve of the whole population (Figure 4.4 [R]). The convergence on the whole population of the clustering solution is slower and steadier than any best solution. The slow convergence on the whole population of solutions depicts the exploration of search space to find better clustering solutions.

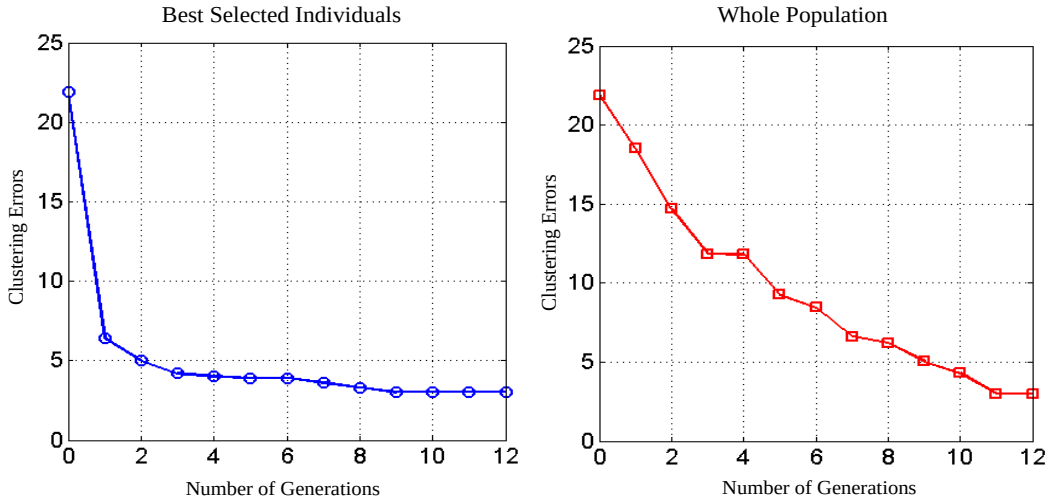


Figure 4.4: Average number of errors (y-axis) obtained at the end of each generation (x-axis), L: Average number of errors obtained for best selected individuals at the end of each generation. R: Average number of errors obtained for the whole population for each generation.

The 2-D projection of the Iris data can be seen in Figure 4.5, with different colors and shapes representing different clusters and solid marks within those shapes showing the memory cells generated. This is a projection of one of the clustering solutions obtained at the end of the algorithm. Fifty-one memory cells were used to store 150 instances, which represents 66.0% data reduction.

For comparison, the standard HAIS algorithm was run 50 times using the Iris data and the results (clustering errors) showed oscillation between good and poor clustering results. The average outcome of these 50 runs was 16.46 errors. The best error obtained was 7, and the worst 27 (see Table 4.2). The population-based HAIS (see Figure 4.4) found better clustering solutions than the standard HAIS for the Iris dataset.

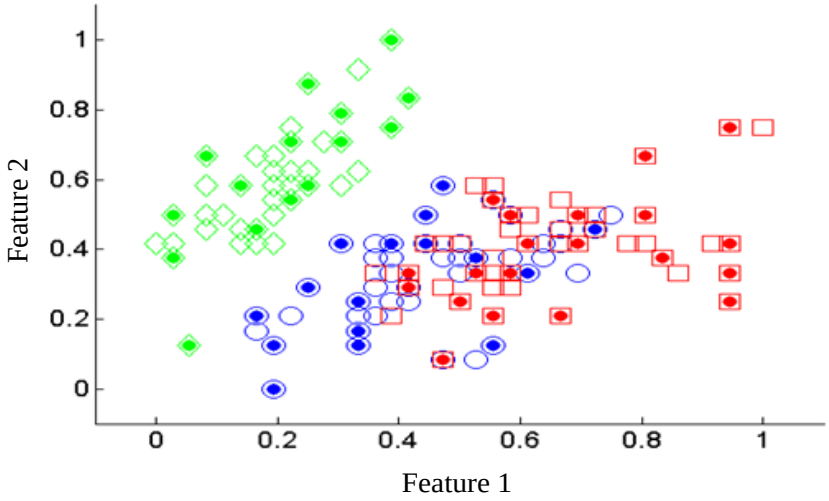


Figure 4.5: 2-D projection of the Iris data using features 1 and 2. Three clusters are shown with different colors and shapes. Memory cells are represented with solid marks.

Table 4.2: Results of the standard HAIS algorithm

Dataset	Min. Error	Max. Error	Avg. Error
Iris	7	27	16.46

Crossover is performed on randomly selected individuals. If no crossover is introduced, the population-based approach can be regarded as simple hill-climbing. An experiment was conducted on the Iris data to justify the use of crossover. The experiment was run 15 times with and without crossover and the average results are shown in Figure 4.6. It can be seen that both curves (best population curve and whole population curve) show the same convergence behavior, but the algorithm with crossover found better solutions in terms of fewer clustering errors. It can also be seen from Figure 4.6 (R) that the convergence curve of population-based HAIS approach with crossover is smoother than without crossover. The minimum, maximum and average results obtained by running this algorithm over 15 runs can be seen in Table 4.3.

The main emphasis of this chapter is to demonstrate that Ag presentation order can affect the HAIS clustering results by keeping all other parameters the same. Different values for parameter AT can produce different clustering results. Therefore, in Step 2 of the algorithm, β and γ of 0.25 and 0.75 respectively were used to achieve fast convergence in the AT parameter value. The values shown in Figure 4.7 are also

averaged over 15 runs using the Iris data. It can be seen from Figure 4.7 (L) that there is not much fluctuation in the AT parameter among different generations. Three AT values representing three clusters in the data are shown using different colors. Figure 4.7 (R) shows the number of memory cells produced against the AT parameter. A clear trend can be seen: as the value of AT, which is a similarity measure, decreases, the final population of memory cells increases and vice versa. Three curves representing the three clusters and their respective ATs and number of memory cells generated for the Iris data are shown in Figure 4.7.

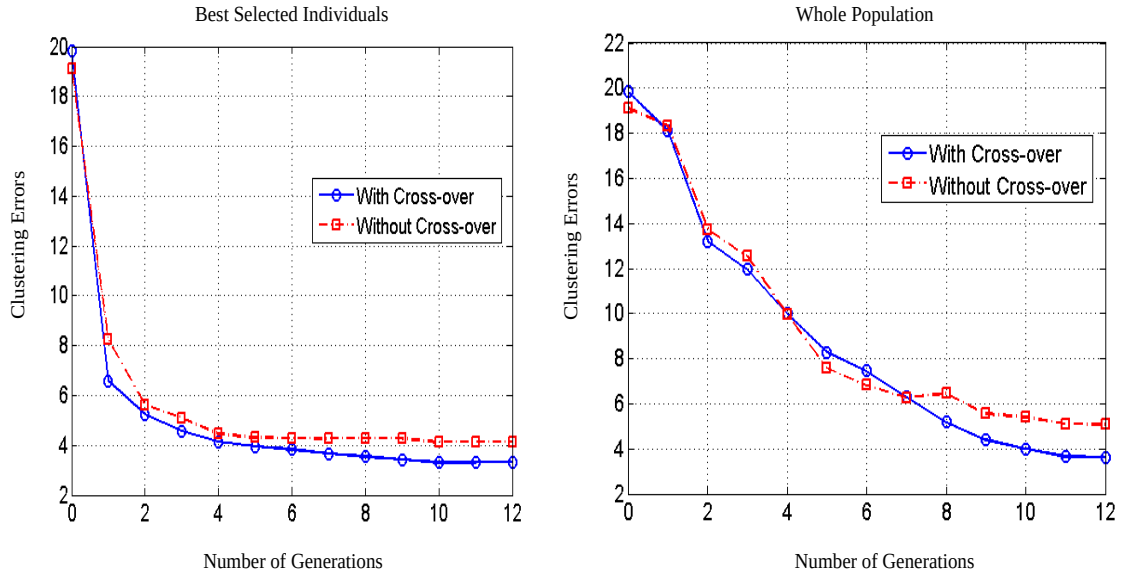


Figure 4.6: Average number of errors (y-axis) obtained at the end of each generation (x-axis). L: Average number of errors obtained for the best selected individuals at the end of each generation, with and without crossover. R: Average number of errors obtained for the whole population for each generation, with and without the crossover operator.

Table 4.3: Results for the best population obtained using population-based HAIS, with and without crossover operator, for 15 runs

Index	Min. Error	Max. Error	Avg. Error
With crossover	3	4	3.267
Without crossover	3.7	5	4.147

The average error over the best population selected at the end of each generation is shown in the Figure 4.8 (L) for four of the dataset datasets (Wine, Thyroid, Breast Cancer Diagnostic and Breast Cancer Wisconsin Original). All error curves show the

same convergence behavior, starting from a high number of errors and gradually converging to some stable local optimum. For the Breast Cancer Wisconsin Original data, convergence was achieved in the 9th generation; for both the Wine and Thyroid data, in the 10th generation; and for the Breast Cancer Diagnostic in the 11th generation. It appears that as the number of features and instances increases, the algorithm takes more generations to settle on local optima.

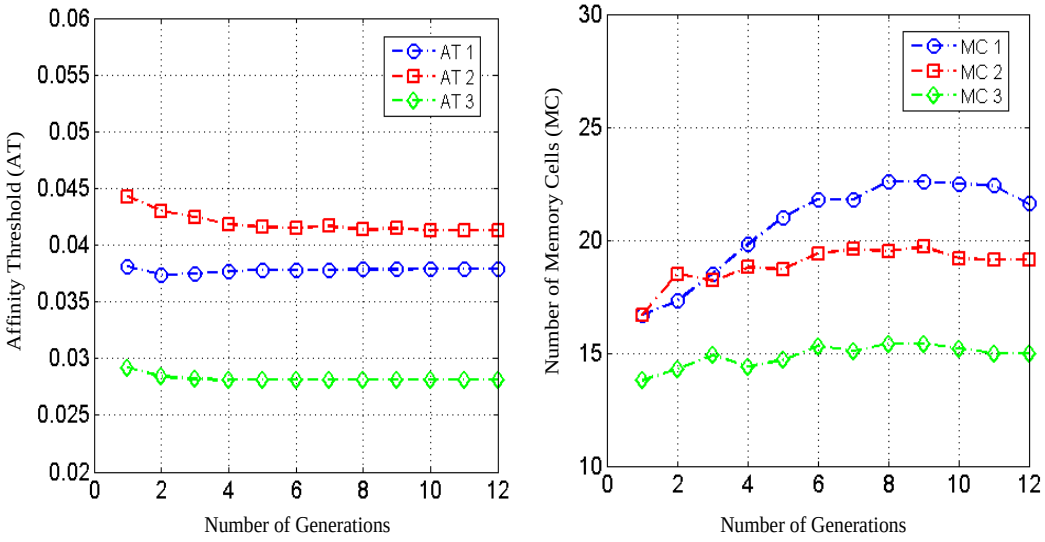


Figure 4.7: L: Average AT for all three clusters of the Iris data obtained at the end of each generation over 10 best selected individuals. R: Average number of memory cells produced at the end of each generation over the 10 best selected individuals.

The average error over the whole population at the end of each generation is shown in Figure 4.8 (R) for the same four datasets. The population error curve also converges as the number of generations increases. This convergence behavior in both the best selected population and the whole population is due to the reinforcement incremental learning of memory cells. That is, the number of memory cells transferred to the next generation increases through the generations to provide a cumulative reward for the system through exploration and exploitation of the environment. The final local optima obtained by the algorithm are dependent on population size and incremental learning through transfer of memory cells. The experiments also show that sometimes the best population and whole population do not converge to the same error, due to differences in population size as well as the presence of a crossover operator. The population-based HAIS algorithm was run for 10 times on the above-mentioned datasets, and the results in terms of minimum, maximum and average errors achieved are shown in Table 4.4.

An interesting observation in Figure 4.8 (R) is that the average errors in the first generation are higher than the average errors in the initial population (population at index zero). In the initial population, the standard HAIS algorithm is used which runs until the termination condition is achieved whereas in the first generation only one Ag presentation order (one shot) is used to obtain clustering solutions. This explains why the average population error in the first few generations is higher than the initial population. However, as the number of generations are increased, as well as the transfer of memory cells from one generation to another, the average errors also followed decreasing error trends and finally settled at the local optima.

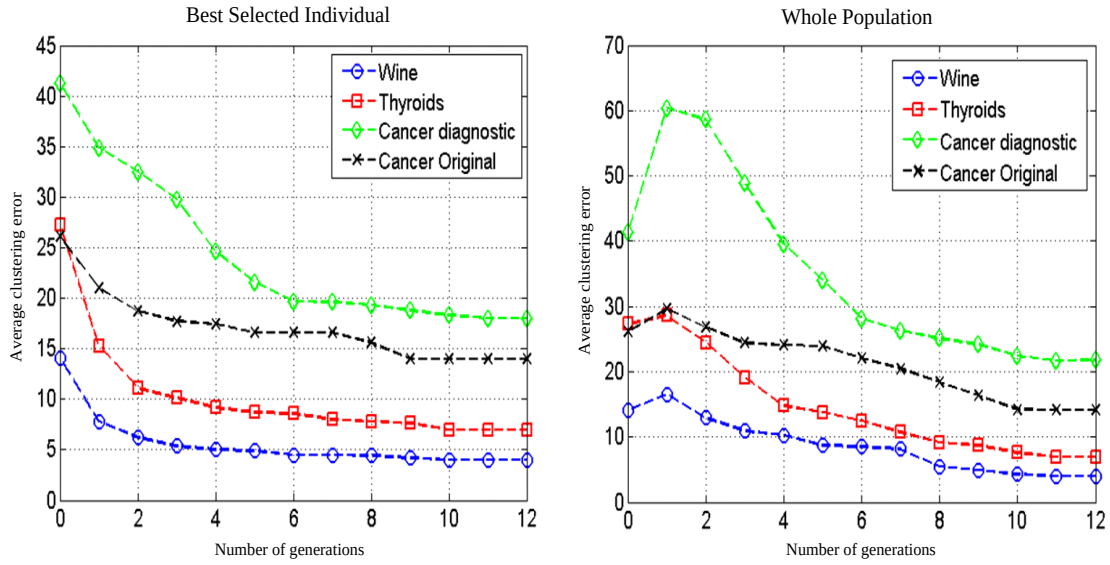


Figure 4.8: Average number of errors obtained (y-axis) in each generation (x-axis) for Wine, Thyroid, Breast Cancer Diagnostic and Breast Cancer Wisconsin Original datasets. L: Average number of errors obtained for best selected individuals at the end of each generation. R: Average number of errors obtained for whole population for each generation

Table 4.4: Average results over 10 runs using population-based HAIS algorithm on benchmark real-world datasets

Index	Min. Error	Max. Error	Avg. Error
Iris	3	4	3.2
Wine	3	5	4.0
Thyroids	6	9	7.2
Breast cancer (diagnostic)	16	21	18.2
Breast cancer (original)	13	16	14.1

4.5 Summary

The main focus of this chapter was to integrate the micro-level processes of an AIS with a macro-level process of a population-based approach to deal with the problem of variable Ag presentation order, which has been found to affect the behavior of the original HAIS algorithm. A population-based approach was incorporated with the standard HAIS that made use of different Ag orders of presentation to achieve local optimal clustering solutions. Section 4.1 covered the motivation and rationale for using a population-based approach. A brief summary and overview of the standard HAIS algorithm was provided in section 4.2. A population-based HAIS algorithm and its detailed explanation were presented in section 4.3. In addition, an adaptive version of HAIS algorithm was also presented, which was important for the proposed population-based HAIS algorithm. The experimental results were presented and discussed in section 4.4. The experimental results demonstrated that the HAIS can benefit from a population-based approach to achieve better clustering outcomes. In addition, crossover was shown to give better clustering solutions than simple mutation of Abs for the datasets used in the experiments here (see Figure 4.6).

In a population-based HAIS algorithm, reinforcement learning is implemented at both micro and macro levels to aid exploitation and exploration of the search space with incremental rewards. Selecting only the fittest (best) individuals forms a basis for macro-level reinforcement learning and incremental reinforcement learning (incremental transfer of memory cells) is performed at the micro-level by selecting only the most affine memory cells. In addition, it has been shown that the population-based approach is not sensitive to data presentation order, as many different Ag presentation orders are used to construct global solutions. That is, the number of errors across runs is stabilized through transfer of the best clustering solutions (as well as memory cells) to the next generation as a result of the elitism strategy adopted. Finally, while 12 generations were sufficient for convergence in our chosen datasets, other datasets may require more generations, means more subtle transfer of memory cells between generations.

In summary, we have shown that Ag presentation order can vary in AIS models and that integrating micro-level AIS processes with macro-level GAs can lead to a benefit that the HAIS by itself cannot achieve. There is a need to investigate the relationship

between hypermutation and crossover to determine closer coupling of micro- (hypermutation) and macro- (crossover) level processes in the context of reinforcement learning. In particular, the relationship between different mutation rates and their effects on optimal allocation of samples to cluster needs to examined. Finally, further work is required to evaluate different population sizes in relation to data size, crossover strategies and numbers of generations for convergence. This chapter has been focused on the importance of Ag presentation order for achieving better clustering results and the experimental results in section 4.4 have clearly shown the importance of this. If different Ag presentation orders can be utilized for effective clustering, then the following questions needs to answered:

1. What is the role of the AT parameter?
2. Can we remove the AT parameter and assign each new Ag according to its closest match to Abs?
3. What is the role of mutation (hypermutation)?
4. Does mutation play any part in achieving effective clustering?

Chapter 5

The Role of Affinity Measure and Hypermutation in the HAIS Algorithm

In recent years, several artificial immune system (AIS) approaches have been proposed for unsupervised learning [11, 102, 130, 131, 136]. Generally, in these approaches antibodies (or B-cells) are considered as clusters and antigens are data samples or instances. Moreover, antigens are trapped through free-floating antibodies or immunoglobulins. In all these approaches, hypermutation plays an important role. Hypermutation is the process of producing cloned but mutated copies of stimulated antibodies to capture similar antigens with high affinity. We presented one such algorithm in chapter 3, namely the Humoral-Mediated Artificial Immune System (HAIS) that is inspired by the role of immunoglobulin in the adaptive immune system. In the previous chapter, we investigated the effects of antigen presentation order in the HAIS algorithm by introducing a population-based approach. However, there is currently little understanding about the effectiveness of hypermutation operator in AIS approaches. In this chapter, we investigate the role of the hypermutation operator as well as affinity threshold (AT) parameters in order to achieve efficient clustering. The

AT in the HAIS algorithm is used to find the degree of similarity between antigen and antibodies. Based on this local similarity measure new antigens are then assigned to either existing B-cells or new B-cells are formed to mount an appropriate immune response. Here, we propose a three-step methodology to examine the importance of hypermutation and the AT parameter in AIS approaches to clustering using HAIS algorithm. The clusters' starting point and antigen presentation order are kept fixed to evaluate the effectiveness and functionality of the hypermutation operator. In addition, the role of mutation in under-fitting and over-fitting the data will be discussed.

The rest of the chapter is structured as follows. Section 5.1 explains the rationale and motivations for adapting a three-step methodology to evaluate AT and hypermutation parameters in the HAIS. Section 5.2 describes in detail the proposed three steps: (1) initial clustering; (2) finding antigen presentation order; and (3) effects of different mutation rates. Section 5.3 presents the experimental results obtained using benchmarked real-world datasets. The role of AT and hypermutation in HAIS is also discussed in terms of entropy in this section. Finally, section 5.4 summarizes this chapter.

5.1 Introduction

Clustering is one of the most intensively researched areas in the unsupervised learning and data mining disciplines. Clustering seeks to group similar data into clusters (groups) so that data instances within a group have maximum similarity while instances across different clusters have a high degree of dissimilarity. Clustering also depends on the nature of the data and the desired results or intuition [1]. Therefore many clustering algorithms exist which use different induction principles. Recently, researchers have turned to natural phenomena for inspiration to develop new clustering algorithms. Underpinning this interest is an inclination with the nature and the emergence of complex learning behaviors and intelligence out of unstructured, unsupervised and decentralized processes, such as those in natural immune systems (NISs).

Biological evolution started billions of years ago and has produced highly complex and efficient organisms. Evolution is, according to the dominant paradigm, driven by natural selection given the constraints of entropy law. Entropy is a measure of the disorder in a system. In simple terms, higher entropy means a higher degree of disorder or

randomness in a system and vice versa. Energy efficiency, processes of recycling, and rewards for co-operation, diversity and decentralization are some of the main characteristics of natural systems [176]. Many nature-inspired clustering algorithms have been proposed in recent years, and the AIS is one such example.

One of the key components of AIS clustering algorithms is hypermutation, which is the process of generating cloned but mutated copies of antibodies (Abs) so that antigens (Ags) can be captured and dealt with more effectively by the immune system. This process maximizes the chances of producing an Ab that is even closer in approximation to the Ag, resulting in: (1) a more efficient handling of the current Ag; and (2) better capturing capabilities for future Ags that could be variants of currently captured one. The biological mechanism underlying this process is referred to as 'affinity maturation'. Most of the AIS clustering algorithms assume that mutation (hypermutation) plays an important role in deciding class memberships of instances of data. One such example is the HAIS described in chapter 3. HAIS is inspired by the role of immunoglobulins (Igs) and Abs in the humoral-mediated response triggered in NISs. However, despite the importance of hypermutation in AIS approaches to clustering, there is relatively little understanding of its effects on clustering algorithms. The initial starting point of clustering (the initial set of Abs and the order in which data samples (Ags) are presented can be expected to play an important role in AIS clustering solutions, along with hypermutation. In the last chapter, we discussed the effects of different Ag presentation order using a population-based approach. In this chapter, the initial clustering starting point and Ag presentation order are kept fixed to evaluate the effects of different mutation rates on the algorithm and therefore shed some light on how hypermutation can affect clustering solutions. Here, we propose a three-step methodology to evaluate the effectiveness of hypermutation in the HAIS clustering algorithm. In Step 1, a hierarchical clustering method is used to get initial data partitioning. The centroids of each cluster are then used as a starting point for the HAIS algorithm. Step 2 evaluates different Ag presentation orders based on the initial starting point (obtained in Step 1). Different affinity threshold (AT) parameters produce different Ag presentation orders. The Ag presentation order that gives the best clustering is selected and used in the third step. Finally, in Step 3, the initial starting point and Ag presentation order selected in the previous steps are respectively used to observe the effects of different mutation rates. Different mutation rates are used to demonstrate the effectiveness of the mutation

operator. Consequent over-fitting and under-fitting by the mutation operator will also be discussed in this chapter in the context of the HAIS clustering approach.

5.2 Proposed Three-Step Methodology

For our AIS, the similarity measure based on the affinity between Ag features and Ab feature receptors is calculated using the following normalized Euclidean expression:

$$D = \frac{1}{f}\sum_{i=1}^{f} |A_i - B_i| \qquad (1)$$

where f is a number of features in the data and A_i and B_i are the absolute values of the Ag and Ab features respectively.

The methodology is divided into three steps. In Step 1, initial clustering is obtained using hierarchical clustering. Step 2 is performed to select an Ag presentation order, ideally giving the lowest clustering error without using any mutation. Finally, Step 3 is used to perform mutation at different rates to achieve a better clustering solution on the Ag presentation order obtained in Step 2.

5.2.1 Step 1: Initial Clustering

In generative clustering approaches, a good starting point plays a very important role in finding good groupings in the data. The model-based Gaussian hierarchical clustering algorithms approach [138], which is a well-established hierarchical model-based method, is used for the initial partitioning. In this method, at each step in the algorithm pairs of clusters are merged so as to maximize the likelihood function $f_k\,(X_i|\mu_k\,, cov_k)$ [177, 178]:

$$f_k\,(X_i|\mu_k\,, cov_k) = \frac{\exp\left[-\frac{1}{2}(X_i-\mu_k)^T\, cov_k\,(X_i-\mu_k)\right]}{(2\pi)^{d/2}\;|cov_k|^{1/2}} \qquad (2)$$

where *X* represent the data, μ_k and cov_k denote the mean and covariance respectively of the k^{th} cluster.

Three different clustering algorithms, namely hierarchical Euclidean distance, K-means clustering, and hierarchical model-based clustering algorithms were tested for the initial partition of the data, from which the hierarchical model-based approach was chosen on account of its superior performance (see appendix C).

5.2.2 Step 2: Finding Antigen Presentation Order

A variant of the HAIS was used to demonstrate the effectiveness of the algorithm using datasets where numbers of clusters and class membership of instances are already known. HAIS is a stochastic algorithm, so that different runs can produce different outcomes. A detailed description of the HAIS algorithm can be found in sections 3.2 and 3.3. The motivation for Step 2 is to get an established Ag presentation order that gives better clustering outcomes based on minimum error against true class labels. Furthermore, in this step no mutation is used, which means that Abs are true data instances without any variation. This step is helpful in investigating the effects of different AT parameters in the HAIS algorithm.

With the number of clusters and initial starting points now being obtained from hierarchical clustering, the third layer of the original HAIS algorithm that deals with generations of new clusters was removed and replaced with a temporary storage unit (TEMP) which is shown in Figure 5.1. TEMP only stores Ags that cannot be trapped at an existing affinity value (AT). These Ags are subsequently put back into the Ag Pool for re-selection after updating the AT parameter.

Step 2 of the algorithm operates on two layers (see Figure 5.1). At the first layer (Layer 1) any memory cell Abs (m-Abs) that already exist trap Ags that are similar as they are released from the Ag Pool and present them to their respective B-cells ([3] in Figure 5.1). The second layer (Layer 2) gets activated if the first layer fails to trap the Ag. At this second layer, the stimulated B-cell Abs (b-Abs) bring the captured Ag to its respective B-cell and also perform certain actions ([5] in Figure 5.1), such as generating a memory cell which results in the generation of further m-Abs, generating k numbers of b-Abs and also performing a negative clonal selection on the newly generated b-Abs. The stimulation level of Ag to m-Ab is set 10 times higher than that of b-Abs to Ag. If the second layer fails to capture the Ag, it is stored at TEMP ([7] in Figure 5.1), which later dumps all Ags back to the Ag Pool ([8] in Figure 5.1). After each successful capture, the algorithm goes back and randomly selects another Ag from the Ag Pool ([6] in Figure 5.1). If there is no selection of an Ag for one complete cycle, the AT parameter is increased appropriately (discussed in next section). This whole process is repeated until all Ags are captured by B-cells.

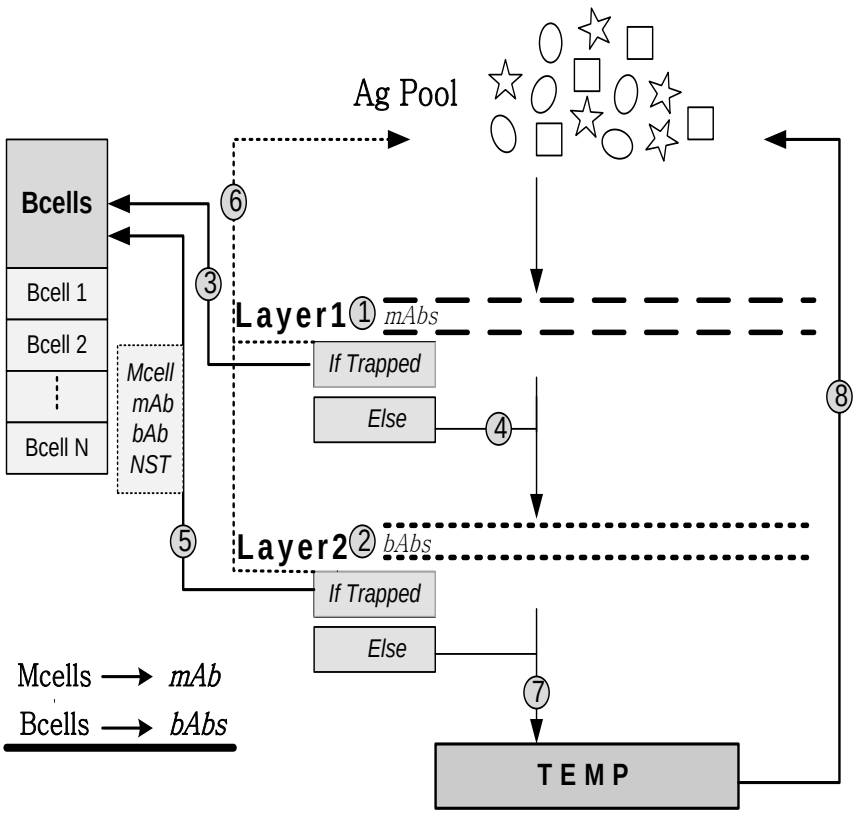


Figure 5.1: Overview of the two-layered algorithm

The above algorithm is highly dependent on the AT parameter, which controls the convergence of the algorithm. The algorithm starts with a small value of AT and, as the immune system matures, the value of AT is increased by some pre-defined quantity to attract and capture Ags with relatively less similarity. This parameter will be discussed in depth in the next section.

In Step 2, no mutation (0% mutation) is used, which means that m-Abs and b-Abs are exact copies of captured Ags. This stage is used to select the Ag presentation order that gives the best clustering results without the influence of a mutation operator. This AT increment continues until all Ags are allocated to their respective B-cells (clusters). The importance of this step is to obtain the best Ag presentation order. Then, in Step 3, the same Ag presentation order is used to allocate Ags to respective B-cells using different mutation rates.

5.2.3 Affinity Threshold (AT) Parameter

In clustering, data instances that are situated closer to the centroids of the clusters should be assigned first and the data instances further away can be left until the size of

the cluster gradually increases and those instances come within range of the cluster. Each cluster can acquire only the data instances that are in its calculated neighborhood radius. Once there are no more data instances within that radius, the radius is increased by some pre-defined quantity to capture the data instances which were out of reach earlier. The same idea is used here in this algorithm, which initially subjects Abs to a very high AT value to get stimulated in order to capture Ags that are highly similar to the Abs. As the clustering process progresses, the affinity threshold is reduced to capture Ags with less, but still enough, similarity to warrant inclusion in the cluster.

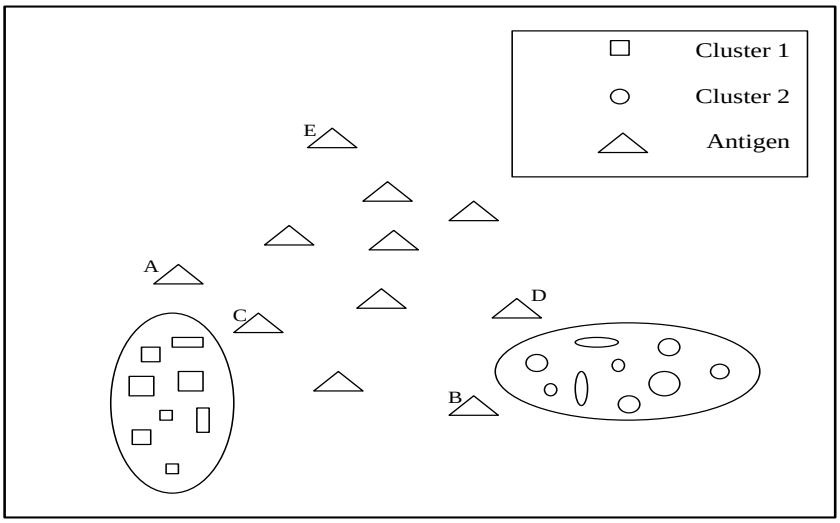


Figure 5.2: Importance of distance measure in cluster analysis

Consider the data instances/points in Figure 5.2. In the case of the cluster containing rectangular-shaped data, Cluster 1 should commit to Ags A and C first. The same principle applies to Cluster 2, which should first acquire Ags D and B rather than committing to Ag E, which is located quite far from both clusters and at this point can belong to either. The initial value of the AT is defined by the expression:

$$AT = \left(\frac{1}{N}\sum_{i=1}^{f} s(X_i)\right) * \alpha \qquad (3)$$

where s is the standard deviation and X_i is the i^{th} data feature, f is the number of features, and N is the total number of instances in the data.

Standard deviation is the square root of variance and is a widely used measure of dispersion and variability. Sums of standard deviation calculated across all the features

represent the total variation or dispersion in a dataset, which when divided by the total number of instances gives the average dispersion of an instance and generates a small starting value of AT. This AT does not stay the same but it changes (increases) as the algorithm proceeds. AT increases when no data instances are captured by Abs in one complete cycle and there are still Ags left to be captured.

The updating of AT can be done in many ways. The simplest is to increase the AT_{inc} by a fixed amount each time when necessary. That technique is not very effective on sparse data, where the algorithm will be forced to perform many redundant cycles. Another approach is to increase it dynamically, and this is the approach adopted here. The AT is incremented by:

$$AT_{inc} = \min\ (affinity) + \left(\frac{1}{N}\sum_{i=1}^{f} s(X_i)\right) * \beta \qquad (4)$$

where *affinity* is the distance measured between b-Abs and Ags and β is another user-defined parameter.

AT_{inc} has two parts: the first finds the next closest distance from Abs to Ags and the second is the initial AT (Equation 3) value with different increasing factor (β), which is to ensure that more than one instance gets selected in the next cycle. The parameters α and β control the rate of convergence. If the values of α and β are set too high then the algorithm converges too quickly. Similarly, if the value is set too low, the algorithm may not converge quickly enough.

5.2.4 Step 3: Effect of Different Mutation Rates

The initial cluster centroids (starting points) were obtained from Step 1, and Step 2 provided a fixed Ag presentation order. Now, in Step 3, a one-shot approach is used to allocate all Ags to B-cells, in the same order as that obtained in Step 2. Many runs can now be performed using variable mutation rates, with a fixed Ag presentation order and fixed initial clustering. Finally, different mutation rates were used to find and evaluate different clustering solutions.

The only difference at this stage from Step 2 is that there are only two layers are activated and any selected Ag must be assigned to one of these two layers, and no TEMP is used (Figure 5.3). At this level b-Abs are generated with different mutation

rates to evaluate the effects of different mutations while generating Abs to capture Ags. Fifty runs (cycles) were performed for each mutation rate. The mutation rates used varied from 5% to 50%. We allowed each feature to mutate within a given specified upper and lower limit. Mutation was designed in such a way that each selected feature could mutate both ways (positively and negatively) at random, rather than just in one direction.

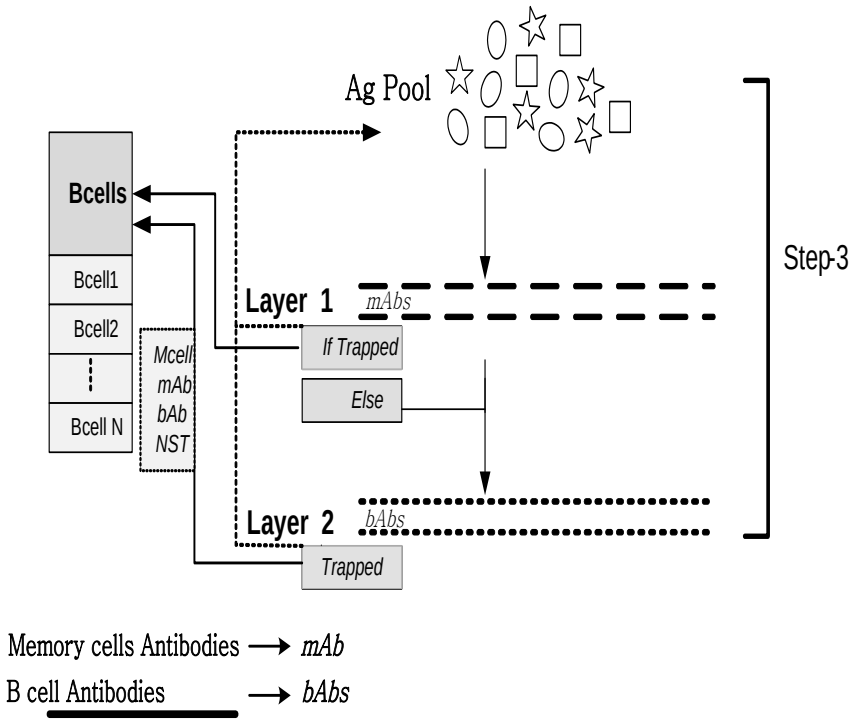


Figure 5.3: A one-shot modified HAIS algorithm using fixed B-cell centroids and fixed Ag presentation order.

In Step 3, the algorithm starts by generating a number of B-cells. The number of B-cells is equal to the known number of clusters in the data. Those B-cells are placed at the centroids of the clusters obtained by Step 1. The B-cells then capture *g* of the nearest neighbor Ags (data instances). The value of the *g* parameter depends on the number of clusters and total number of instances in the data. Now, these B-cells generate *k* numbers of cloned and mutated copies of captured b-Abs to capture similar Ags in the future. The *k* Abs produced are dependent on two main factors. Firstly, the number of features in the data: the higher the number of features, the higher the number of Abs generated should be to find good enough samples to attract similar Ags. Secondly, the mutation rate: the higher the mutation rate, the higher the number of Abs should be to ensure sufficient diversity in the system. This *k* parameter directly influences the time

the algorithm takes to run. The Ag captured by each B-cell is considered the memory cell, and this stimulated memory cell will generate respective m-Abs with a very small mutation rate to capture similar Ags. For the experiments done in this chapter, the m-Abs are exact copies of captured Ags.

5.3 Experimental Results and Discussion

Four well-known datasets, namely Iris, Wine, Thyroid and Breast Cancer Wisconsin (see appendix A) were used to show the effectiveness of the proposed methodology. Each of the datasets selected shows different degrees of data complexity, which demonstrates the feasibility of the methodology. The hierarchical clustering was performed on normalized data whereas Steps 2 and 3 were performed on raw data except for the Wine data, where normalized data was used in all three steps.

When the hierarchical model-based clustering algorithm was used on the Iris data, it produced 15 clustering errors. The confusion matrix obtained can be seen below:

	Group1	Group2	Group3
Cluster 1	50	0	0
Cluster 2	0	49	1
Cluster 3	0	14	36

In Step 2 (with variable Ag presentation orders), the algorithm was run for 50 times, each time with different α and β parameters. The results obtained using different α and β parameters can be seen in the Figure 5.4 (without mutation). The lower values of α and β mean fewer ways to present Ags or less randomness in the Ag presentation order. On the other hand, the higher values of α and β mean more ways to present Ags to the algorithm.

In the legend of Figure 5.4, AB21 denotes α and β values of 1 and 2 respectively. It can be observed from Figure 5.4 that with AB21 the number of clustering errors oscillates between 4 and 6. The second-best set of parameter values was obtained by using AB22, which gave a minimum of 6 clustering errors. Out of the α and β parameters listed in Figure 5.4, it can be seen that AB21 consistently gave better clustering results (meaning fewer clustering errors) without using any mutation rate. Figure 5.4 also shows the importance of different data presentation orders for clustering solutions given AT

parameter, which was the basis of using a population-based approach in a chapter 4. Figure 5.4 demonstrates that a lower AT parameter produces less fluctuations in terms of variations obtained in clustering results, whereas a higher AT parameter produces higher fluctuation is clustering results.

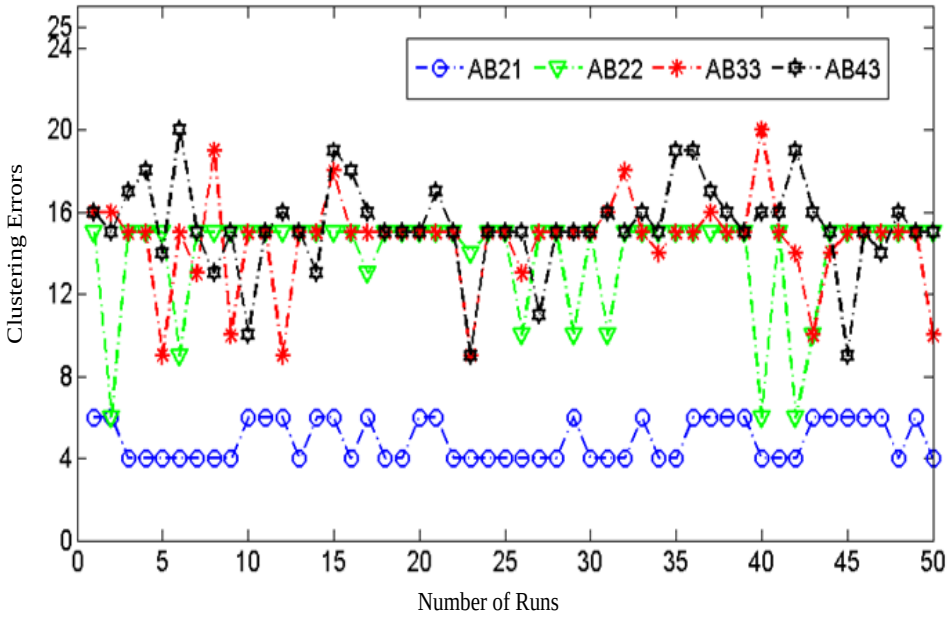


Figure 5.4: : Iris data clustering errors using various α and β values with 0% mutation (Step 2), with the x-axis representing the number of runs and the y-axis the number of clustering errors obtained.

A summary of the experimental results given in Figure 5.4 in terms of minimum, maximum and average clustering errors found using different α and β parameters is given in Table 5.1. These results suggest that AB21 has the least errors and that its error average is also the lowest. The worst average of clustering errors of 15.288 was obtained from AB43. The experiments were also performed using other values of α and β, but only four are shown here. The purpose of conducting these experiments is to show that even without using mutation, α and β influence the final clustering solution and that the AT parameter plays an important role in AIS clustering algorithms. Another aspect of these results is that running the algorithm for 50 times with the same AB21 and the same initial clustering mean produces different clustering outcomes. This underlines the importance of the way Ags are presented to the system for selection (Ag presentation order). The purpose of this three-step methodology is to fix the cluster starting point as well as the order of presentation in order to observe the effects of different mutation rates on the clustering outcome.

Table 5.1: Summary of Figure 5.4

	Min. Errors	Max. Errors	Avg. Errors
AB21	4	6	4.8846
AB22	6	15	13.75
AB33	9	20	14.577
AB43	9	20	15.288

To explain the behavior of mutation rates, the Ag presentation order showing 15 clustering errors was selected (Step 2). Here, a fixed Ag presentation order was used that has been obtained by using AB22 parameter values (see Figure 5.4). The mutation rates of 5%, 10%, 15%, 20%, 30% and 40% were used and each was run 50 times, as shown in Figure 5.5. Different colors and shapes represent different mutation rates, defined in the legend of Figure 5.5. A summary of Figure 5.5, in terms of minimum, maximum and average clustering errors, can be seen in the Table 5.2. When the mutation rate was 5%, the oscillation was minimal but as the mutation rate increased the oscillation in the clustering solutions increased also. The clustering error without mutation was 15, therefore 15 can be considered as the mean point or point of reference for this experiment. It can be seen that for mutation rates of 5%, 10%, 15% and 20%, the oscillation is around the mean point, whereas in the case of mutation rates of 30% and 40% the clustering errors are much higher and away from the mean point. More specifically, with a mutation rate of 5% the error curve is not too far from 15 (the average is 15.67 in Table 5.2).

The best and worst clustering solutions were 14 and 18 errors, respectively, for the 5% mutation rate. However, with a 10% mutation rate the oscillation becomes higher and it had best and worst clustering solutions of 5 and 19 errors respectively. The lowest clustering error of 4 was found with a mutation rate of 15%. The clustering results gradually degrade from this point onwards for all the remaining mutation rates. This experiment suggests that mutation plays an important role in finding better clustering solutions. When the mutation rate is kept too low, it cannot generate a diverse enough population of Abs to match the Ags as they are introduced to the system; as a consequence the results are not much different from 0% or no mutation. On the other hand, in the case of too high a mutation rate the Abs cover too much feature space, which results in bad clustering solutions. Abs not only represent already seen samples

but also ‘predict’ future samples that should belong to the same cluster. Hence, low mutation rates suffer from under-fitting when predicting future samples based on existing samples, whereas too high a mutation rate cause over-fitting.

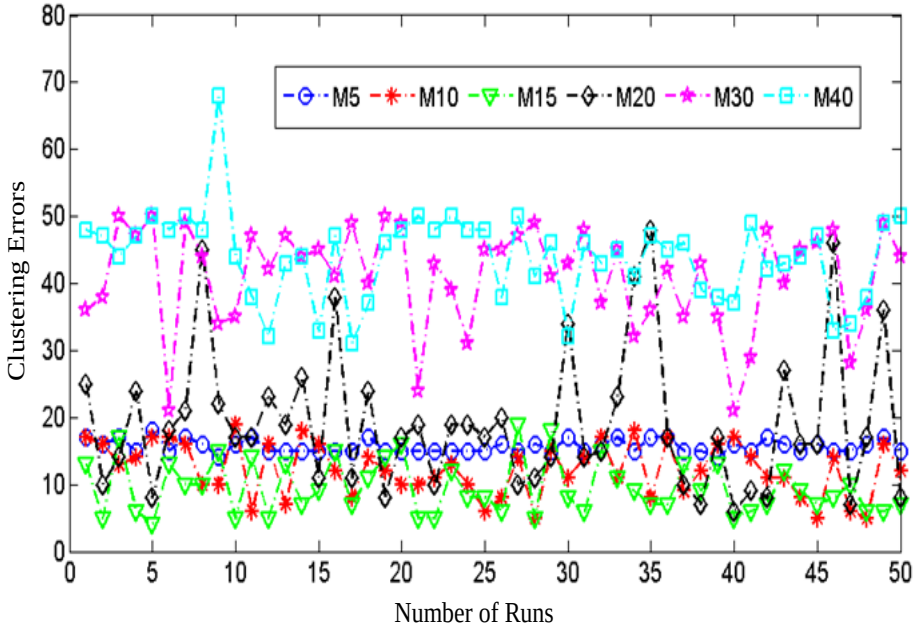


Figure 5.5: Various mutation rates M5 to M40 (5% to 40%) are applied to an Ag presentation order with 15 clustering errors

Table 5.2: Summary of Figure 5.5

Mutation	Min. Errors	Max. Errors	Avg. Errors
M5	14	18	15.67308
M10	5	19	12.23077
M15	4	19	9.596154
M20	6	48	19.48077
M30	21	50	40.82692
M40	31	68	44.21154

Using α and β values of 2 and 1 respectively returned the minimum number of clustering errors (4) in the previously explained experiment. The same Ag presentation order can be used to investigate the effects of mutation rates on clustering error. In particular, it will be possible to determine whether different mutation rates using the Ag presentation order show the same behavior as in Figure 5.5 (using an Ag presentation order with 15 clustering errors). The results obtained using mutation rates of 5%, 10%,

15%, 20% and 30% can be seen in Figure 5.6. The same trend can be seen as for the previous experiment (with 15 clustering errors). The role of mutation can now be clearly seen in Table 5.3, which is a summary of Figure 5.6. A minimum clustering error of 2 was found when a mutation rate of 15% was used as compared with 4 clustering errors without mutation. This experiment suggests that including mutation rate in the HAIS clustering algorithm does provide extra benefit in terms of reducing clustering error, if it is tuned correctly.

The 2-D projection of the final clustering on the Iris data can be seen in Figure 5.7 (L), where each cluster is represented by different colors and symbols. The 2-D projection of the final Abs (b-Abs) obtained is also shown in Figure 5.7 (R).

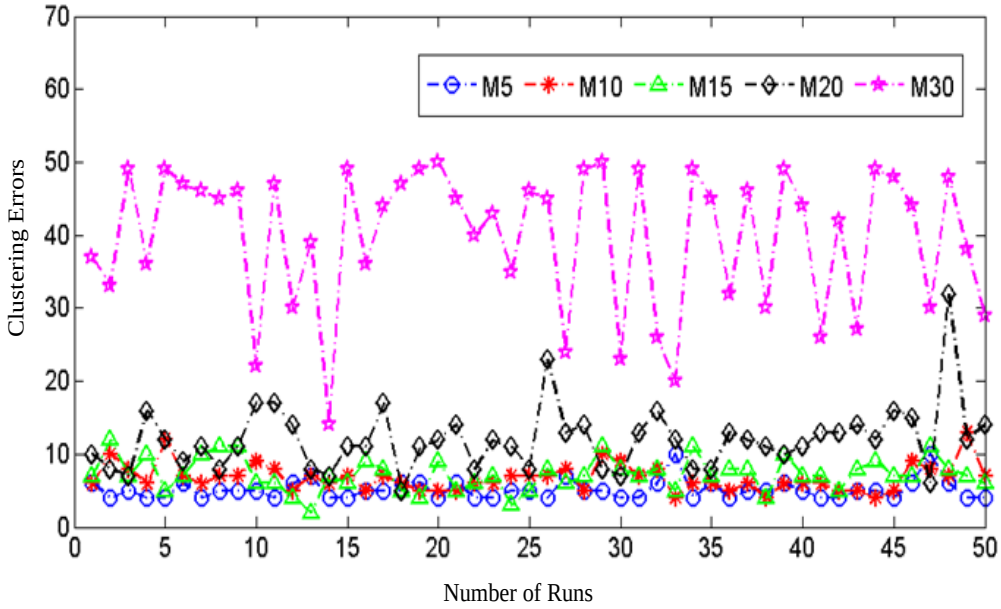


Figure 5.6: Various Mutation rates M5 to M30 (5% to 30%) are applied to an Ag presentation order with 4 clustering errors

Table 5.3: Summary of Figure 5.6

Mutation	Min. Errors	Max. Errors	Avg. Errors
M5	4	10	5.173077
M10	4	13	6.788462
M15	2	12	7.230769
M20	5	32	12.26923
M30	14	50	39.42308

The negative clonal selection threshold (NegT) plays an important role in the HAIS as it deletes similar Abs and creates empty spaces (in terms of feature space) that subsequently could be occupied by the same or different class of Abs. Reinforcement learning can also be performed at the NegT parameter level depending on whether negative selection is performed at the generation of Abs level or at the B-cell. If negative selection is performed at the level of Abs creation, the same class of Abs can also occupy the empty feature spaces in the future (i.e. after further Ag presentation). This would not be possible if negative clonal selection is performed at the B-cell level because then only Abs from different classes can occupy the gaps in the future. In this chapter, negative clonal selection at the level of generation of Abs is performed to allow reinforcement learning in terms of generating Abs.

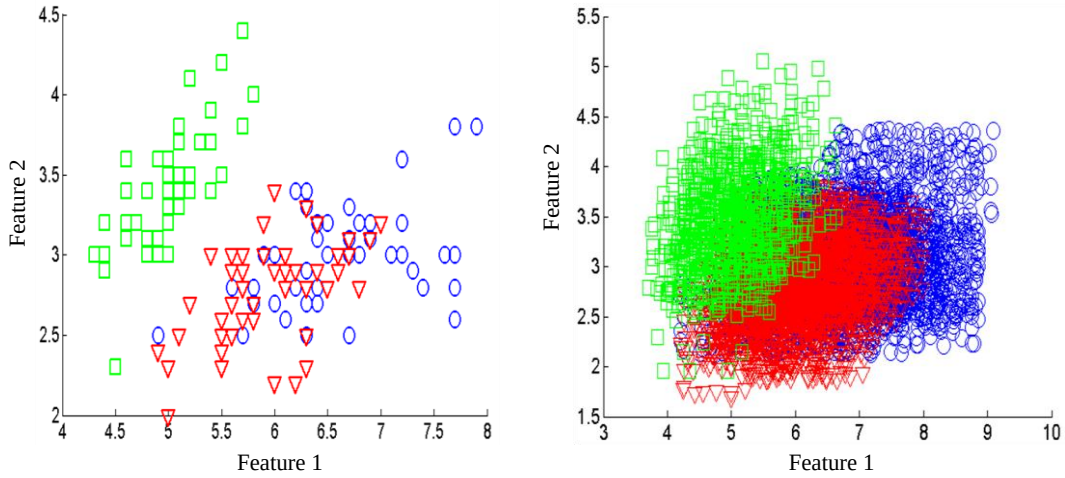


Figure 5.7: L: Three clusters of the Iris data obtained at the end of the algorithm. R: Abs produced at the end of the three-step algorithm with 15% mutation rate.

The best results found using the proposed methodology on all datasets can be seen in Table 5.4. The α and β parameters were obtained manually, by trying various combinations, while 50 runs were performed against each mutation rate to get the best results.

The comparison of clustering errors found at the end of each step is shown in Table 5.5, which suggests improved results at Step 3 from both the previous steps. Note that in the results obtained at the end of Step 3 (after applying sub-optimal AT and mutation rates) for all four real-world datasets, the final clustering results are superior only when applying the hierarchical clustering algorithm (Step 1). These results were obtained using only 50 runs in Step 3; much better results can be obtained if Step 3 is repeated

for more iterations (explained below). One of the reasons for the observed oscillations at Step 3 is the use of intra-cluster B-cell negative selection (negative selection of Abs for each cluster or B-cell separately) and allowing Abs to enforce natural selection pressure across all clusters. This approach was applied under the assumption that clusters can overlap in Euclidean space, which forms the basis of using intra-cluster (and not inter-cluster) B-cell negative selection. All these fluctuations (in clustering results) at the level of AT and hypermutation in Figures 5.4, 5.5 and 5.6 can also be explained using the norm of entropy.

Table 5.4: Datasets information (three-step methodology)

Datasets	α	β	**Mutation**	**Min. Errors**	**Max. Errors**	**Avg. Errors**
Iris Data	2	1	15%	2	12	7.23
Thyroids Data	15	5	40%	11	44	22.19
Wisconsin Data	15	7	50%	17	29	23.09
Wine Data	3	2	15%	6	22	13.26

Table 5.5: Step-wise errors information

Datasets	**Step-1**	**Step-2**	**Step-3**
Iris Data	15	4	2
Thyroids Data	23	20	11
Wisconsin Data	23	29	17
Wine Data	7	9	6

The various mutation rates can also be explained in terms of the entropy of a system. Entropy is defined as a number of ways the constituents of a system can be re-arranged, in such a way that a change would not be noticed [179]. Different systems can vary from a low entropy state to a high entropy state. Low entropy means there are fewer possible arrangements existing (the system is more organized), whereas in a high entropy state many possible arrangements exist (less organized system). Entropy can be explained using the simple example shown in Figure 5.8, where (a) shows some randomly located gas molecules in a box. The system is considered to be a closed system and gas molecules move around in the box. There are many ways the system in (a) can re-arrange itself so that no difference is noticeable. Therefore, it can be said that

the system in (a) is in a high entropy state. On the contrary, the system in (b) is said to be in a low entropy state as there are fewer arrangements existing. According to Sean Carroll [179], a low entropy state can be achieved from a high entropy state, but it takes a much longer time. Therefore, if the system in high entropy (a) is left for a long time, due to the random motion of gas molecules all the configurations attainable will be attained and it is possible that the molecules with re-arrange themselves into the configuration shown in (b), which is a low entropy state. Entropy is defined as:

$$E = k \log(N) \qquad (5)$$

where $N =$ number of states and k is the Boltzmann's constant.

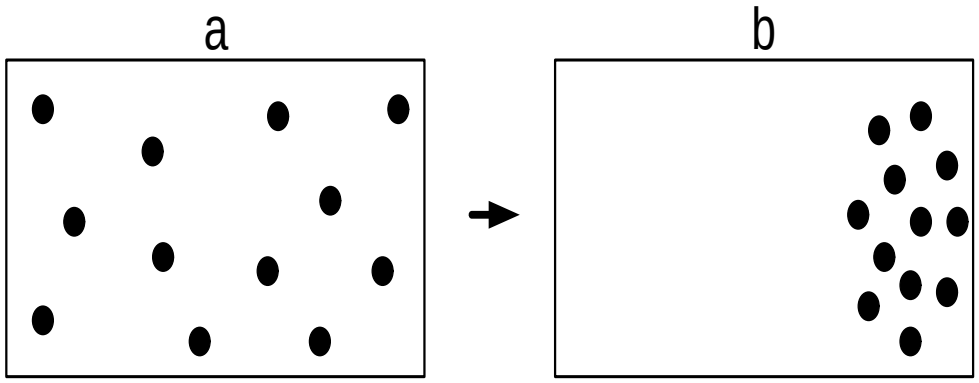


Figure 5.8: Fluctuations in a closed system (gas molecules) occasionally lead from a high entropy state (a) to a low entropy state (b)

Now the question requiring an answer is: How can clustering be related to entropy? Clustering algorithms can mainly be classified into deterministic and stochastic algorithms. In deterministic algorithms, different runs of the same algorithm using the same data produce the same clustering results. One example of such algorithms is hierarchical clustering algorithms (Step 1, section 5.2.1). On the other hand, stochastic algorithms produce variable clustering results across different runs. Therefore, in the light of the entropy definition explained earlier, deterministic algorithms can be categorized into systems (algorithms) having a low or zero entropy state and stochastic algorithms, which oscillate between different clustering solutions, can be regarded as systems having a high entropy state. The HAIS algorithm is a stochastic algorithm, meaning it can produce different clustering outcomes on various runs, which suggests it has high entropy. The important question here is to consider what are the most important factors or parameters in the HAIS that make it such a stochastic algorithm.

In this chapter, the two most important features of the HAIS are investigated, namely the AT parameter and hypermutation. The results shown in Figure 5.4 indicate that a low AT is equivalent to a low entropy state and a high AT equivalent to a high entropy state. A low AT parameter such as that of AB21 has only two possible arrangements (4 and 6 clustering errors), but as the AT parameter value is increased, the system showed many more possible arrangements (more fluctuations in clustering results). The minimum errors obtained on the Iris data at AB21 and AB43 were 4 and 7 respectively when only 50 runs are used. Now, an interesting experiment would be to see whether a system with higher AT (e.g. AB43) that is left for a longer time (more runs) randomly fluctuates to a low entropy state that corresponds to a low clustering error configuration. The experiment was run using AB43 for 1000 iterations. The results can be seen in Figure 5.9 below and suggest a higher level of fluctuations (meaning high entropy) in the clustering results. The four clustering errors earlier obtained using AB21 were obtained many times in the 1000 iterations using AB43, which proves that a low entropy state can be obtained from higher entropy state but, due to the random fluctuations, it can take a longer time. Another important feature of this experiment was that it achieved a minimum of 3 clustering errors, which also demonstrates the HAIS's capacity to find better clustering solutions while exploring wider search space by increasing the AT parameter. This experiment also showed that there is a trade-off between local search and global search for finding optimal clustering solutions. AB21 was consistently able to find 4 and 6 clustering errors, but even though AB43 on average found fewer efficient clustering solutions, it did find a clustering solution with 3 errors, i.e. better than those of AB21. Therefore, we can characterize algorithms (or algorithmic settings) that produce more fluctuations as systems with high entropy, and vice versa. The same behavior can be observed in hypermutation. As the rate of mutation is increased, the system is transformed from a low entropy state to a high entropy state (see Figure 5.10). An Ag presentation order of 4 clustering errors was used to demonstrate this idea. The same setting was run for 1000 iterations using 5% to 30% mutation rates. A 15% mutation rate has found the lowest number of clustering errors (clustering error of 1).

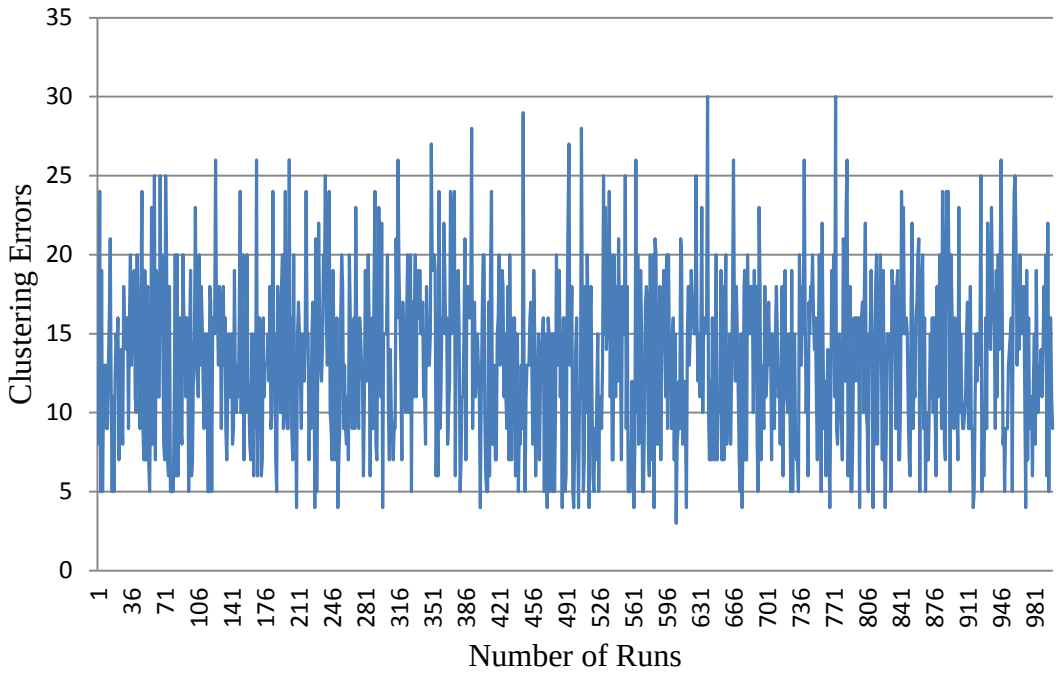
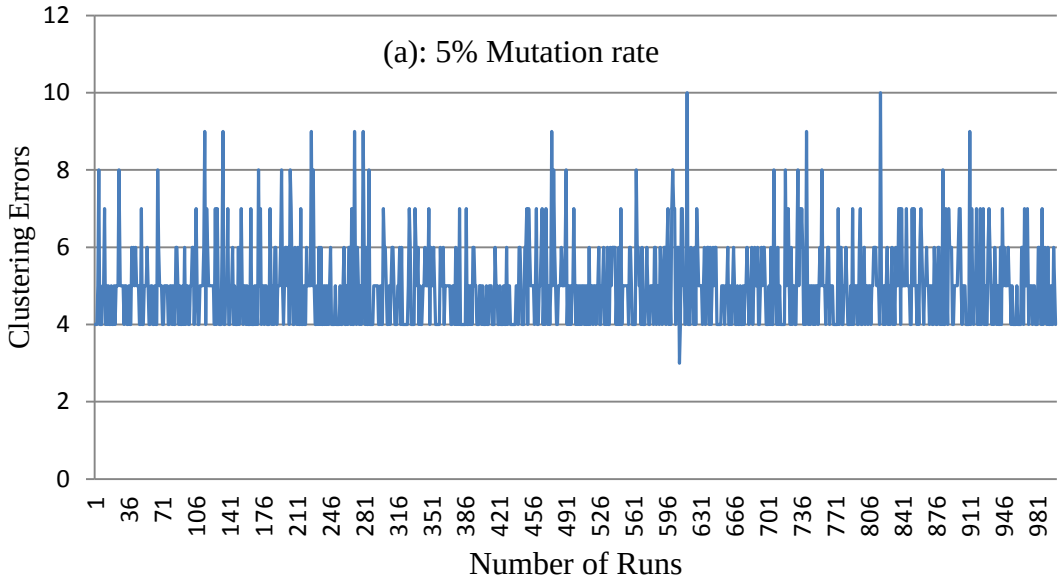
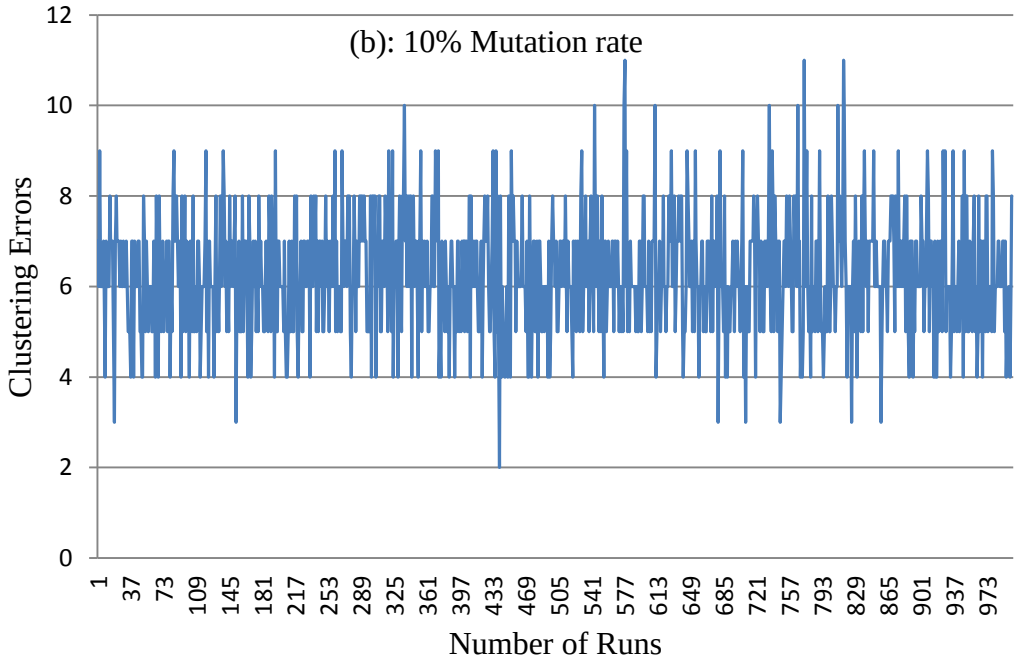


Figure 5.9: Iris data clustering errors using AB43 (Step 2). The x-axis represents the number of runs and the y-axis the number of clustering errors obtained.

Here we are interpolating the idea of the number of possible states *N* (which is proportional to the entropy value *E*) to the range of clustering solutions (fluctuation in clustering errors) obtained. In these experiments, *N* is equivalent to AT as well as *N* is equivalent to mutation rate.

From the experiments conducted here, it can be concluded that a low AT or low mutation rate describes a low entropy state and a high AT or high mutation rate characterizes a high entropy state.


(a): 5% Mutation rate
Clustering Errors
Number of Runs
0
2
4
6
8
10
12
1
36
71
106
141
176
211
246
281
316
351
386
421
456
491
526
561
596
631
666
701
736
771
806
841
876
911
946
981


(b): 10% Mutation rate
Clustering Errors
Number of Runs
0
2
4
6
8
10
12
1
37
73
109
145
181
217
253
289
325
361
397
433
469
505
541
577
613
649
685
721
757
793
829
865
901
937
973

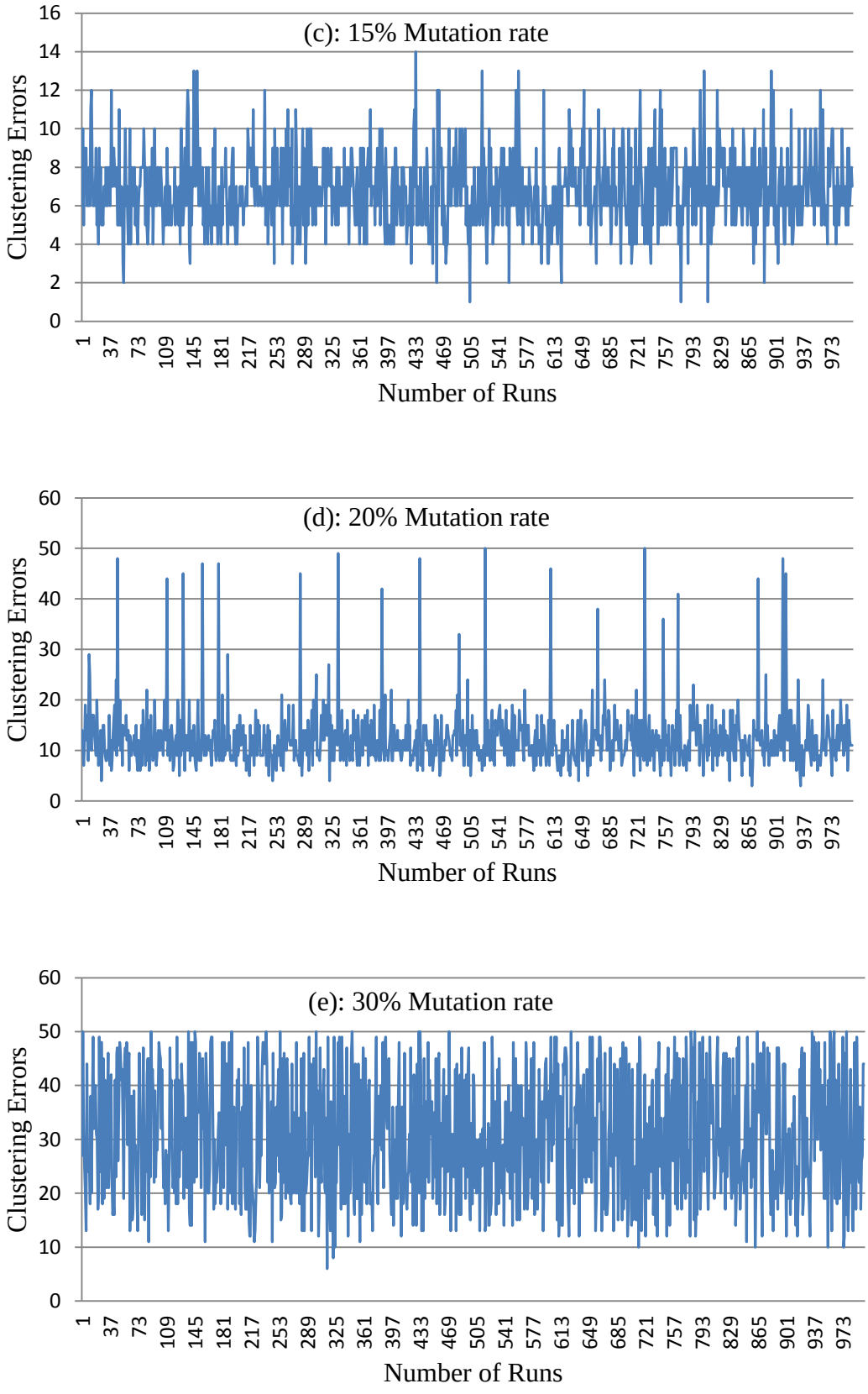


Figure 5.10: Various mutation rates (5% to 30%) with 4 clustering errors at Step 2. The x-axis represents the number of runs whereas y-axis the number of clustering errors obtained.

5.4 Summary

Both affinity measure (AT) and hypermutation play critical parts in any AIS algorithm. The aim of this chapter was to determine the effects of both parameters on unsupervised clustering while using the HAIS algorithm. AT is a similarity criterion that is very important in defining the boundaries of Ab receptors (in terms of shape space) and the vicinity they interact in (put another way, AT helps to distinguish self, known and seen search space from unknown, non-self and unseen search space). Hypermutation is another critical component of the AIS and getting the mutation rate right is important for effective Ag-Ab match and capture.

There are three main parameters that can influence the clustering outcomes in the HAIS algorithm: (1) initial B-cell placement; (2) similarity measure (AT); and (3) mutation rate. The main focus of this chapter was to investigate the effects of different affinity measures and mutation rates on the HAIS algorithm. We have demonstrated with experimental results on real-world datasets that both AT and mutation play an important role in finding better clustering solutions. The experimental results indicate that mutation, along with the AT parameter and the initial B-cell starting point, helps the HAIS algorithm produce better clustering results. The implications are not clear for other AIS approaches to clustering given that they have other parameters. Nevertheless, the role of mutation can be expected to play a key part in other AIS approaches to clustering. For example, the aiNet and CLONALG algorithms use similarity thresholds to select strong affinity Abs. These similarity thresholds act in a similar manner to our mutation rate. We have also tried to map the behavior of two core parameters of the HAIS algorithm, namely AT and mutation, onto entropy. In this chapter we have argued that a low AT or mutation rate can be linked to a low entropy state and a high AT or mutation rate to a high entropy state.

So far, we have demonstrated the capabilities of the HAIS with regard to unsupervised learning. In the coming chapters, we will focus on developing a variant of the HAIS algorithm that can be applied to supervised learning.

Chapter 6

Humoral Artificial Immune System (HAIS) for Supervised Learning

Two critical natural mechanisms in evolutionary processes are variation and selection and these form the basis of what is called evolutionary computing (EC). EC has proved successful when dealing with complex problems, such as classification, clustering and optimization. In recent years, as our knowledge of microbiology has deepened, researchers have turned to micro-level biology for inspiration to help solve complex problems, and one such example is the natural immune system (NIS). In previous chapters, we have presented the Humoral-Mediated Artificial Immune System (HAIS) algorithm which was inspired by the humoral-mediated immune response for unsupervised clustering and outlier detection. This chapter extends this idea and describes a novel supervised learning algorithm inspired by the humoral-mediated response triggered by the adaptive immune system. The proposed HAIS supervised

learning algorithm uses core immune system concepts such as memory cells, plasma cells and B-cells, as well as parameters and processes inspired by our knowledge of the microbiology of immune systems, such as negative clonal selection and affinity thresholds.

The rest of the chapter is structured as follows. Section 6.1 explains the motivation and introduction of the novel supervised HAIS algorithm proposed in this chapter. Section 6.2 presents a literature review of existing AIS supervised learning approaches. Some NIS concepts employed in our new proposed AIS algorithm are discussed in section 6.3. Section 6.4 then describes the novel AIS algorithm for supervised learning and an explanation of the algorithm is presented in section 6.5. Extensive experimental results conducted on benchmarked real-world datasets are presented in section 6.6. Parameters and processes used in the HAIS supervised algorithm are explained in section 6.7, and a discussion regarding the HAIS supervised learning algorithm forms section 6.8. Finally, a summary of the chapter is presented in section 6.9.

6.1 Introduction

The goal of any supervised learning algorithm is to build a model that can accurately predict unseen data instances by learning from existing patterns consisting of predictor features and class labels.

Supervised learning is one of the most frequently used techniques in the machine learning domain. A large number of techniques based on various induction principles can be found in the literature. Supervised learning techniques are generally classified into the following five main groups [180].

1. Logic-based algorithms, including symbolic learning methods. Examples include decision trees and classification rules. The most well-known algorithm for building a decision tree is the C4.5 [37], which is an extension of the ID3 algorithm [181]. RIPPER is a well-known example of a rule-based algorithm [182]. An extensive overview of various rule-based algorithms can be found in [183].
2. Neural networks or perceptron-based techniques. These approaches range from single-layered perceptrons to multi-layered perceptrons and redial-basis function networks [184]. Single-layer perceptrons consist of only two layers namely, input and output layers, whereas multi-layer perceptrons consist of an input layer and an

output layer with an additional hidden layer [15, 18, 185]. An overview of existing work in artificial neural networks can be found in [13, 41, 186].

3. Statistical learning algorithms. These models are built on the basis of probability modeling of the data. Examples are Naive Bayes classifiers [39] and Bayesian networks [38, 187].
4. Instance-based learning algorithms are also called lazy-learning algorithms [34] in which the induction or generalization process is delayed until classification is performed. The k-nearest neighbors (k-NN) method is an example of this group of algorithms. A review of instance-based learning algorithms can be found in [188].
5. Support Vector Machines (SVMs) are a relatively new addition to supervised machine learning algorithms [189]. 'SVMs revolve around the notion of a "margin"—either side of a hyperplane that separates two data classes' [180]. SVMs construct a hyperplane (or hyperplanes) in a high-dimensional space that is later used for classification tasks. A survey of SVMs can be found in [40]. More details regarding these supervised machine learning groupings can also be found in [180].

Nature-inspired techniques for classification have their roots in our early and macro-level understanding of biological processes. The contribution of various nature-inspired techniques such as of artificial neural networks (ANNs), genetic algorithms (GAs), ant colony optimization (ACO) and particle swarm optimization (PSO) is well established in machine learning. However, as our knowledge of biology has deepened, another trend has emerged in nature-inspired systems, which is to go further into the microbiology. This trend is based on the view that if nature has found 'algorithms' for solving complex problems, where such algorithms have arisen because of particular types of bio-molecular machinery we possess, we may be able to inform our own computational algorithms by including aspects of such machinery in their design and development. In other words, hardware, or in this case 'wetware' (artificially created cellular processes), matters when it comes to biological inspiration. A relatively recent example of deeper biological inspiration is the AIS. What distinguishes an AIS from other nature-inspired techniques is its assumption that going deeper into the biology may provide new computational paradigms not obvious if one remains at the macro level.

AIS researchers are therefore interested in the behavior and functionality of AISs due to the fact that they can address some important questions that other classifiers also have to deal with, such as:

1. Has the particular pathogen/data sample been encountered before?
2. How certain are we about the identity/class of a certain pathogen/data sample?
3. Do the pathogens/data samples share features that could be useful for identification/classification?

The aim of the work presented in this chapter is to develop a new AIS supervised learning algorithm which can enhance our understanding of patterns in the data. The proposed algorithm incorporates the latest knowledge of how the humoral-mediated parts of the NIS work. This chapter also addresses, with a novel perspective, issues of over-fitting and under-fitting of data while building an AIS classifier. The novelty of the proposed algorithm is discussed in the context of an existing immune system based on supervised learning algorithms. The performance of the proposed algorithm is tested on well-known benchmarked real-world datasets and the results indicate performance no worse than existing techniques in most cases and improvements over previous reported results in some.

6.2 Literature Review

Some work has already taken place in AIS supervised and unsupervised learning algorithms [11, 89, 126, 190]. The fundamental principle is to represent data samples as antigens and classes/clusters as antibodies or B-cells so that antigens are attracted to those antibodies/B-cells that are most similar. Similarity is typically measured through standard metrics, such as the Euclidean distance between the attribute values of a sample and the centroid of a class or cluster of samples, with samples being allocated to the closest class or cluster. Other standard metrics for calculating similarity (e.g. squared Euclidean distance, Pearson correlation) between samples and dissimilarity (e.g. between groups, within groups) between classes/clusters can also be used.

CLONALG [74], a clonal selection algorithm originally specified for binary character recognition and engineering optimization, has also been adapted for unsupervised clustering. Recently, a revised version of CLONALG, called CLONAX [191], was presented that was designed for supervised learning and the obtained results were

compared against other benchmarked techniques. CLONAX is functionally very similar to CLONALG. Many of the processes such as affinity calculation, cloning of antibodies, and affinity maturation are similar in both approaches. However, there are also a few differences in these approaches. Firstly, the user can define the size of the final set of memory cells obtained by the CLONAX algorithm. Secondly, CLONAX evolves memory cells separately for each class of antigens with the objective of minimizing classification errors on seen data or antigens. Finally, CLONAX performs noise filtering by considering memory cells from the opposite class while building a final model. The k-NN approach is applied to evaluate the performance of generated memory cells.

The main work on the AIS supervised learning algorithm is the Artificial Immune Recognition System (AIRS) of Watkins *et al.* [88], which uses the concepts of artificial recognition balls (ARBs), resource limitations, memory cells and hypermutation. ARBs are essentially B-cells supplemented with information on the resources available in the system. AIRS adopts a one-shot approach in that learning patterns (antigens) are allocated to the closest matching ARB in the pool of ARBs, followed by a competitive stage, where the ARBs either survive or die depending on their fitness with regard to capturing antigens of the right class. Resources are re-allocated throughout the ARBs depending on which ARBs survive or die. Memory cells are produced from the surviving ARBs. At the end of the one-shot approach, the memory cells (two or more of which can represent one class) adopt a k-NN voting method by presenting test samples to all memory cells and reporting the stimulation values returned by each memory cell.

AIRS adopts several immune system concepts and the results reported are competitive with other traditional supervised learning algorithms. However, there is nothing to stop the number of ARBs, and hence memory cells, from proliferating beyond the number of classes of data, which is why a final k-NN approach is required. In addition, while the concept of memory cells is used in AIRS, its functionality is different. In an NIS, once the memory cell is generated by a B-cell, it stays in the body almost indefinitely, but in AIRS memory cells can be replaced by more affine or more active memory cells. In an NIS memory cells form the first line of defense in the recognition of pathogens after affinity maturation so that a faster response to already seen pathogens can be mounted in the future. In the AIRS algorithm, however, pathogens/antigens are not exposed to memory cells at the training stage. Instead, each pathogen goes through a recognition

cycle through ARBs. Finally, the affinity threshold (AT) in AIRS is calculated as the average distance between all pathogens in the system. In AIRS each class (memory cell) has the same AT value, which is based on the assumption that all classes are homogenous in the structure. This may limit the effectiveness of AIRS for other types of real-world data. The AT proposed in the AIRS algorithm can lead the model to under-fit or over-fit the data on this assumption. In other words, AIRS essentially is a cluster-based approach to supervised learning, supplemented with a k-NN stage for allocating antigens and their associated pathogens into classes and a resource-based fitness model to reduce the number of candidate memory cells.

Another critical issue is generalization. The purpose of generating memory cells is not only to represent the whole dataset with fewer instances but also to keep information on the original structure of data and classes. A model (final set of memory cells) which does not truly represent the whole data can suffer from under-fitting. A model that contains memory cells that do not represent the full variety of potential pathogens that belong to that memory cell class can therefore lead to under-fitting. On the other hand, a model that contains too many memory cells to represent pathogens can suffer from over-fitting. Therefore, lack of generalizability to new samples, where the model is a poor representation of the data, can lead to higher classification error, and hence results may not be much better than k-means or k-NN classifiers on test data. These problems of over-fitting and under-fitting can be addressed through the use of appropriate AT measure, as will be seen later.

Nearly all other AIS supervised learning algorithms, as far as we are aware, are based on AIRS or its variants, and include ARBs, some form of k-NN voting, and resource limitation as part of a fitness measure. This chapter presents an immune system-based classifier which is more closely inspired by the NIS. The proposed algorithm works on two layers to capture free-floating pathogens. The first layer consists of memory cells to represent the first line of defense in the adaptive immune system, whereas the second layer is based on antibodies produced by B-cells. The role of negative clonal selection, which is omitted by the AIRS, is also used in our algorithm. Negative clonal selection is used here to generate and keep a diverse population of antibodies in the repertoire at all times. The concepts of local mutation and hypermutation are used by AIRS and are also used in our algorithm to explore wide search spaces. Two similarity measures, namely

global AT (same AT for all classes) and local AT (a different AT measure for each class), are proposed in this chapter to avoid over-fitting.

6.3 Immune System Concepts Employed

In our proposed algorithm, B-cells represent the classes and have unique receptors for the binding of pathogens (samples). The functionality of B-cells is to hold the pathogens until the end of the algorithm as well as produce antibodies once an antigen is presented to a B-cell via other associated memory cells or antibodies. Once an antigen is presented to a B-cell, the process of affinity maturation takes place and the B-cell further divides into two cells: a plasma cell and a memory cell. The plasma cell subsequently produces antibodies, which are hypermutated copies of the matured stimulated antibody, whereas the memory cell is an exact copy of the matured stimulated antibody. This memory cell now stays in the system to capture similar antigens. The interaction between antigen and antibodies or antigen and memory cell is measured using a standardized Euclidean distance calculation: a comparison is made between antigen and antibodies or memory cells and the antibody/memory cell and whichever shows the highest affinity (match) captures the antigen. When an antibody captures an antigen, the antigen leaves a genetic blueprint on the antibody, which is called affinity maturation. During this process the stimulated antibody fine tunes its receptors towards the captured antigen.

Negative clonal selection is an important concept in an NIS and is also a key component in our algorithm. This concept is very important for keeping a diverse population of antibodies in the repertoire at all times, as well as for removing redundant antibodies. Negative clonal selection is undertaken against its own class of antibodies. In addition, if two generated antibodies are too similar to each other, only one copy is kept while the other is removed. After generating the final set of antibodies and undertaking initial negative clonal selection, another check is made against the newly created antibodies with already existing antibodies in a specific class. At this stage, if a newly generated antibody is too similar to an already existing antibody, the newly generated antibody is removed from the repertoire. More detail regarding negative clonal selection can be found in the experimental section (section 6.6). Two kinds of mutation are used here in the proposed HAIS algorithm, namely hypermutation and local mutation. Generation of new antibodies is achieved through hypermutation while local mutation is used for the affinity maturation process (fine tuning of the stimulated antibody). Memory cells are

another important component of this algorithm; it is these memory cells that are used for classifying test data. More details regarding memory cells are discussed later in section 6.6. The mapping between NISs and data mining concepts used in this chapter is shown in Table 6.1.

Table 6.1: Mapping of NIS expression to AIS concepts

NIS Expressions	Data Mining Concepts
Antigens/pathogens	Samples
B-cells	Classes
Antibodies	Variations of already existing instances
Antigens to Antibodies interaction	Pair-wise comparison between new and existing instances
Similarity measure	Normalized Euclidean distance
Mutation	Creates diverse population of solutions
Negative Clonal Selection	Controlling antibodies' population
Memory Cells	Capture the essence of already seen patterns
Affinity threshold	Similarity criterion
Affinity maturation	Learning from new patterns

6.4 HAIS Supervised Learning Algorithm

An overview of the proposed HAIS algorithm for supervised learning is now provided (a full explanation follows the algorithm). Abbreviations used are: B-cell; antibodies produced through plasma cells (Abs); affinity threshold (AT); memory cell (M-cell); antigen (Ag); negative clonal selection threshold (NST); affinity maturation threshold (AMT). All the data instances are regarded as Ags; hence the term Ag Pool is used, where all Ags are placed for random selection one at a time.

1) **Set parameters:** AT, Local Mutation, Hypermutation, AMT
2) Start with generating as many B-cells as classes in the data
3) Randomly select a data instance from each class, assign them to B-cells, create M-cells and Abs
4) Run until no Ags left in the pool
5) Randomly pick an Ag from the pool
6) Get Ag's label and brings it to respective B-cell
7) Compare it with existing M-cells in the system and select the most stimulated M-cell
8) ***If*** difference between stimulated M-cell and Ag is less than AT, do:
    i) The respective B-cell captures Ag, and go back to step 5
9) ***Else***
    a) Compare Ag with the existing Abs generated by same B-cell, and select the most stimulated Ab
    b) ***If*** difference between stimulated Ab and Ag is less than AT, do:
        i) B-cell will capture Ag
        ii) Stimulated Ab will undergo affinity maturation process to get its receptor tuned with Ag

iii) The tuned Ab produces Abs
iv) B-cell generates an M-cell
v) Negative clonal selection is performed on all Abs in selected B-cell
vi) Go back to step 5

c) ***Else*** **(***if existing Abs can't capture Ag)*
i) Mutate the most stimulated Ab until the required affinity maturation is achieved
ii) B-cell captures the Ag
iii) M-cell is generated
iv) Generate Abs
v) Negative clonal selection is performed
vi) Go back to step 5

d) ***End if***

10) ***End if***
11) ***End main***

6.5 Explanation of Algorithm

As noted earlier, the algorithm uses normalized Euclidean distance to calculate the similarity between Ags and Abs/M-cells. As an initialization phase, the data is normalized between zero and one to give each feature equal weight, and the distance calculated as:

$$D = \frac{1}{f}\sqrt{\sum_{i=1}^{f}(A_i - B_i)^2} \qquad (1)$$

where D is the similarity measure, f is the number of features (number of columns in the data if Ags are stored in rows), and A_i, B_i are Ab and Ag features respectively.

If the minimum of the similarity measure is less than the AT, then the Ab/M-cell gets stimulated (Steps 8 and 9a, b and c in the algorithm above). In this chapter two separate measures of calculating AT are discussed which can be used depending on the structure of the data. Equation (2) below calculates a global AT for each class (in the data) by considering the sum of the standard deviation of each feature in the data and then dividing it by the number of features in the data. The resulting value is then divided by a user-defined parameter α, to make the affinity measure stronger or weaker among Ags and Abs while calculating similarities.

$$AT = \frac{\frac{1}{f}\sum \sigma\,(data)}{\alpha} \qquad (2)$$

where σ is the standard deviation of the data and α is the user-defined parameter which must be a non-negative and greater-than-zero value.

On the other hand, in Equation (3) each class is assigned a different (local) AT depending on its own standard deviation. In other words, the sum of the standard deviation of data related to each class is calculated separately, and then divided by the number of features to calculate a local AT value for each class. The effects of α on the production of final model (M-cells) is discussed in section 6.6.

$$AT = \frac{\frac{1}{f}\sum_{i=1}^{labels}\sigma(data^i)}{\alpha} \qquad (3)$$

where σ is the standard deviation of the data, and $data^i$ is all data instances associated σ with class label i.

The stimulated Ab brings the captured Ab to the B-cell, which holds it till the end of the algorithm. The B-cell then undergoes a process of affinity maturation and produces a plasma cell and a M-cell. Affinity maturation is the process by which the stimulated Ab fine-tunes its receptors in the direction of the antigenic receptors. This affinity maturation is controlled by another user-defined parameter called the AMT.

$$AMT = \frac{AT}{\beta}, \ \ where\ \beta > 0 \qquad (4)$$

The AT is the value calculated in Equation (2) or Equation (3), and β is a user-defined parameter that depends on the data structure and also on the value of AT.

There are two mutation rates used in this chapter, namely local mutation and hypermutation. Hypermutation is used to cover a wider search space, whereas local mutation is used for the purpose of fine-tuning the Abs to attain affinity maturation. For the purpose of the experiments here, a hypermutation rate of 10% to 20 % and local mutation rate of 5% was used. In short, Abs are created by using hypermutation and they then fine-tune their receptors in the direction of Ags through local mutation.

Negative clonal selection is one of the key ideas used in this algorithm. During the process of producing plasma cells and consequently releasing Abs into the system, negative selection is performed to remove too similar Abs. Negative clonal selection is conducted at two levels. At the first level, when the new Abs are generated, a comparison is made among all newly generated Abs and if two are too similar to each other, one of them is removed. At the second level, just before releasing the newly

created Abs into the system, they are also compared against the already existing Abs in the system (against their own class), and again if two are too similar then the newly generated Ab is removed from the system. Negative clonal selection is performed against its own class of Abs and not against Abs of another class to encourage natural competition.

6.6 Experimental Results

The proposed algorithm was tested on a number of benchmarked datasets to assess the classification accuracy. This section will explain the results obtained on various datasets and then discuss the behavior and functionality of some of the parameters used in this algorithm. The datasets Iris, Ionosphere, Pima Diabetes, Sonar and Thyroid (see appendix A) were used. The algorithm's performance is compared against AIRS and AIRS2 [88], which are well known AIS- based classifiers.

After training the algorithm the final set of M-cells obtained was used for classification, which was achieved using k-NN. Different values of *k* were tried and finally 3-NN was selected (majority voting between M-cells) for all datasets. These results were obtained from averaging over 5 to 10 runs and using different methods of cross-validation. For comparison purposes the same scheme of cross-validation used by Watkins in AIRS [88] was used here. For example, in the case of the Iris data 5-fold cross-validation was employed with averaging over 10 runs on the same data division as reported in [88]. The k-fold schemes of all other datasets are explained in Table 6.2. All the datasets use a k-fold scheme except the Ionosphere dataset, where the first 200 data instances were regarded as training data and the remaining 151 instances as testing data.

The experimental results shown in Table 6.3 suggest that our algorithm performs as well as the AIRS and AIRS2 algorithms, and for Thyroids and Sonar datasets it produces better results.

One of the interesting features of AIS algorithms in general is their capability of data summarization or data reduction through the process of M-cell production. These M-cells are then used to classify any unlabeled data. Table 6.4 presents a comparative analysis of the data summarization of our algorithm against AIRS and AIRS2. The results indicate that HAIS produces equal to or sometimes better than AIRS2 data summarization for the Sonar, Ionosphere and Thyroid datasets, whereas its performance

is relatively poor on the Iris and the Pima Diabetes datasets. However, overall the capability of data reduction of the proposed algorithm is competitive against AIRS2. The parameters used in these experiments are given in Table 6.5. Various parameters were tested; however, only parameter values that gave consistently better classification results are stated.

Table 6.2: k-folds scheme used in this chapter

Dataset	k-folds	Runs
Iris	5	10
Ionosphere	200 Training/151 Testing	10
Pima Diabetes	10	5
Sonar	13	10
Thyroids	10	10

Table 6.3: Comparison of accuracy of the HAIS algorithm against other AIS classifiers using benchmarked data

Dataset	Avg. Acc. (%)	Best Acc. (%)	M-cells	AIRS1 %	AIRS2 %
Iris	96.00	97.33	56/120, 53%	96.7	96.0
Ionosphere	95.36	96.67	99/200, 51%	94.9	95.6
Diabetes	74.13	75.21	315/692, 46%	74.1	74.2
Sonar	86.54	88.46	128/192, 33%	84.0	84.9
Thyroids	96.05	97.21	112/192, 41%	94.82	---

Table 6.4: Comparison of M-cell formation (data reduction)

Dataset	Size	HAIS	AIRS1	AIRS2
Iris	120	53%	65%	74%
Ionosphere	200	51%	30%	52%
Diabetes	691	46%	32%	60%
Sonar	192	33%	25%	7%
Thyroids	194	41%	---	---

Table 6.5: Parameters used in the experiments

Dataset	**Alpha**	**Beta**	**NST**	**Mutation (%)**
Iris	4.5	10	0.02	15
Ionosphere	9	10	0.015	15
Diabetes	5.5	10	0.025	15
Sonar	12	10	0.015	15
Thyroids	5	10	0.025	15

When there are homogenous class structures in the data, it is easy to avoid over-fitting and under-fitting of data while building a predictive model. However, in real-world datasets where classes within a set vary in structure, over-fitting and under-fitting can be very difficult to deal with. This can be explained with the help of the diabetes data taken from [177]. This dataset has three classes (Normal, Chemically Diabetic and Overtly Diabetic). It was selected here to observe the behavior of over-fitting when each class has a different degree of sparseness. The dataset was used to test the effects of local and global AT measures proposed in this chapter, and the results were then compared with AIRS1 and AIRS2.

In Table 6.6, the original data had a total of 145 instances, where 76 instances belonged to class 1 (normal), and 36 and 33 instances to classes 2 and 3 respectively. The global AT in HAIS was able to perform approximately 54% data summarization in comparison to the original data. That is, the variation in 145 samples has been reduced to 67 M-cells, which represents a 54% reduction. But it can be clearly seen that classes 2 and 3 together only achieved 28% of data reduction for a global AT (the average of 23 and 26 M-cells obtained from 36 and 33 original data instances). Furthermore, the same trends can also be noticed when AIRS1 and AIRS2 are used; they over-fit classes 2 and 3. This is in comparison to a local AT, where the total data reduction is about 60% and classes 2 and 3 show almost the same (59%) data reduction. The 3-D projection of original data showing the three classes with different colors and symbols, and the M-cells (black marks) obtained at the end of the HAIS algorithm using a global AT and local AT are shown in Figure 6.1. The evidence of over-fitting for classes 2 and 3 (square- and triangle-shaped classes) can be seen in Figure 6.1. In Figure 6.2, where a local AT measure is used, this issue has been resolved and the data reduction in all three classes is almost equal.

It can be concluded from this experiment that global AT only works well when classes are known to be homogenous. Homogeneous classes are where the data making up the class possess homoscedasticity (the samples making up the class have the same finite variances and distributions across attributes) and are uniform in composition and character. The local AT, which uses variable AT values for each class, can be used to avoid over-fitting or under-fitting issues.

Table 6.6: M-cell formation in diabetes data for each class with data summary capability

	Class1	Class 2	Class 3	Total	Errors
Original	76	36	33	145	-
Global AT	18	23	26	67 (53.7%)	2
Local AT	31	15	12	58 (60.0%)	3
AIRS1	34	27	26	87 (40.0%)	7
AIRS2	23	24	29	76 (47.6%)	2

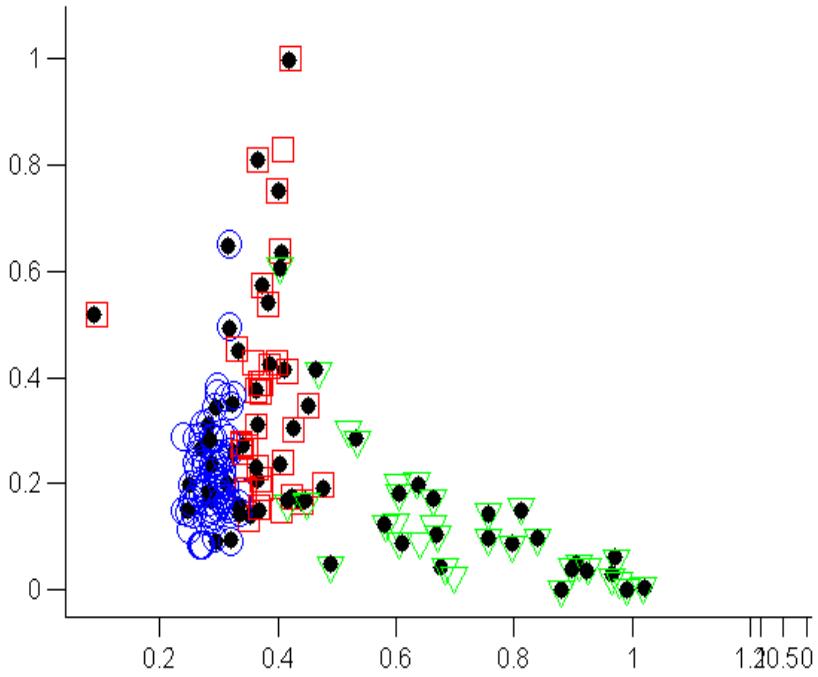


Figure 6.1: 3-D projection of original data and M-cells generated by the HAIS algorithm using a global AT measure

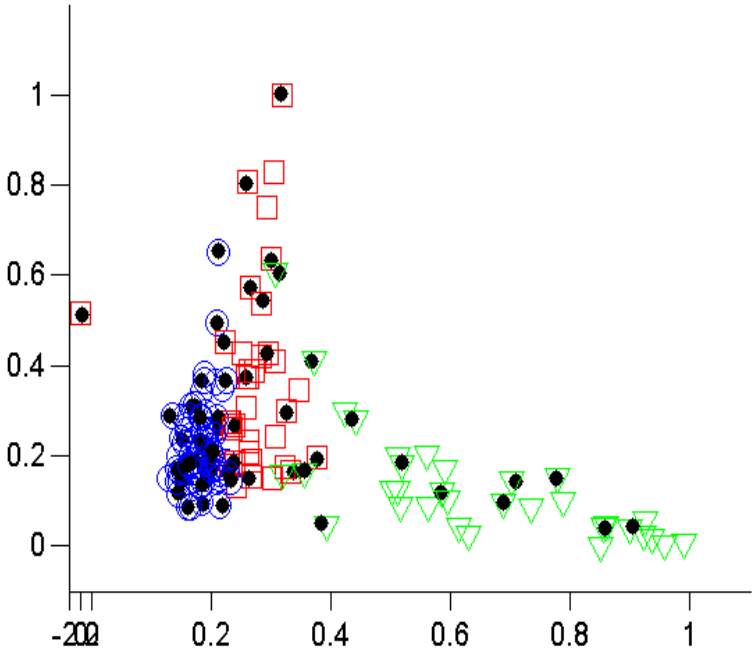


Figure 6.2: 3-D projection of original data and M-cells generated by the HAIS algorithm using a local AT measure

As explained in the previous section, negative clonal selection is very important in controlling the population of Abs. The results of applying a simple negative clonal selection process are shown in Figure 6.3. A simulated data point x = [0.25 0.75] is created and using 10% hypermutation 200 Abs are then generated. Blue circles represent Abs generated using data point x. Then negative selection is performed using various values of NST ranging from 0.005 to 0.020. The final sets of Abs obtained are shown as black spots in Figure 6.3. It is clear that negative clonal selection and especially the NST parameter can be used to obtain diverse and non-similar population of objects. For the purpose of these experiments, NST ranged from 0.010 to 0.020, depending on the dataset used.

M-cells are one of the core components in the HAIS algorithm and are used to classify new instances. Figure 6.4 presents the original 2-D projection of the Iris dataset, and each class is shown using different shapes and colors. Solid-colored marks are the final sets of M-cells obtained at the end of the algorithm. Class-wise allocation of M-cells is explained in the caption to Figure 6.4. The α (for AT) and β (for AMT) are set at 4.0 and 10 respectively. The complete dataset was used here to demonstrate production of the final set of M-cells, which is 71 out of 150 instances. These M-cells can be used for validation or testing data when class labels are missing or hidden.

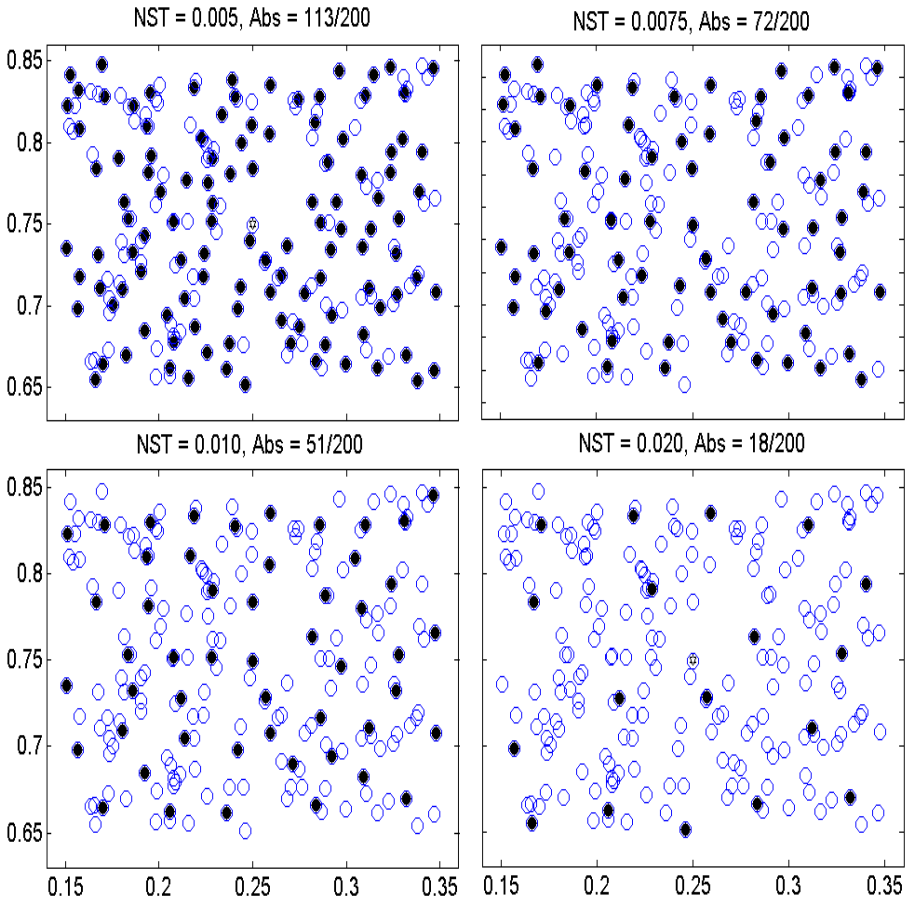


Figure 6.3: Negative clonal selection of Abs using various NST parameters

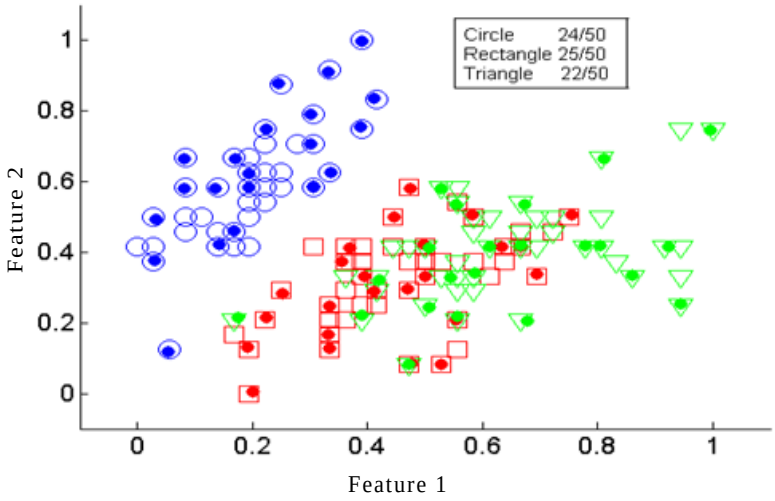


Figure 6.4: Final M-cells produced by the HAIS algorithm using α = 4.0 (M-cells = 71/150)

As mentioned earlier, α controls the stimulation level of Abs/M-cells during the interaction of B-cells and M-cells with Ags. A higher α means that an Ab/M-cell needs a greater level of similarity with Ags for the appropriate B-cell to be stimulated, and

vice versa. The relationship between the final set of M-cells and α can be seen in Figure 6.5, where the number of M-cells show a positive linear relationship with the α parameter. In other words, as the value of α increases, the number of M-cells also increases.

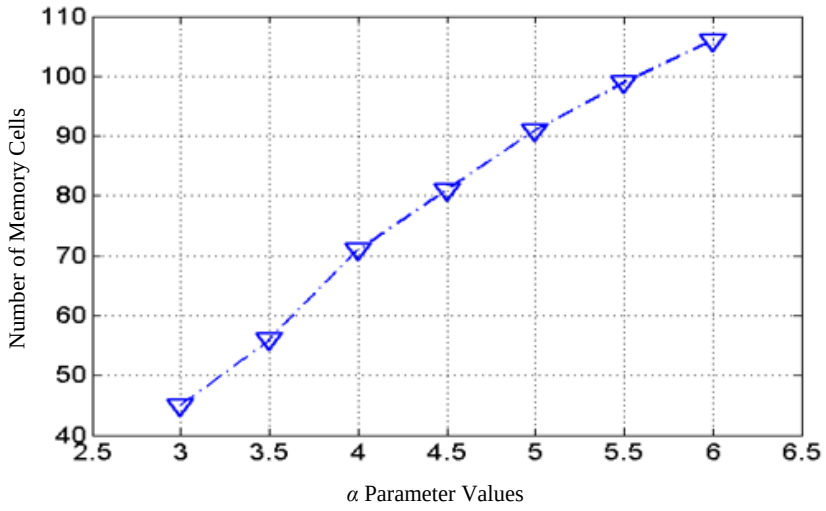


Figure 6.5: Average number of M-cells (y-axis) obtained at various values of α (x-axis)

6.7 Parameters Evaluation

The experiments conducted so far have focused only on the classifier's accuracy on unseen data (test data) and data reduction capabilities on seen data (training data). There has been little attention given to the effects of various parameters such as α and β, or how various Ag presentation orders can affect classification results. The following experiments conducted in this section were designed to serve four main purposes:

1) Highlight the effects of different Ag presentation orders in the HAIS supervised learning algorithm.
2) Illustrate the role of affinity maturation in the HAIS supervised learning algorithm.
3) Demonstrate the effect of the α parameter on classification accuracy.
4) Demonstrate the effect of the β parameter on classification accuracy.

6.7.1 Antigen Presentation Order

The Iris dataset is divided into 5-folds and the same 5-fold distribution of data instances was maintained for all the experiments reported on here. The effects of different Ag presentation orders in achieving better clustering solutions have been discussed for

unsupervised clustering in previous chapters. The supervised version of the HAIS also faces the same challenge, in that its performance is affected by the Ag presentation order (the way M-cells are formulated). The training phase of the HAIS algorithm produces M-cells which can be used in the testing phase to assign labels to unseen and unlabeled data instances. The different Ag presentation order produces different sets of M-cells. This problem is highlighted in Figure 6.6, where different sets of M-cells are generated, hypothetically, based on the different order of Ag presentation.

In the proposed HAIS algorithm, data instances (Ags) are selected purely at random to be captured by their respective B-cells, therefore the position of M-cells are also random in the HAIS algorithm. These sets of M-cells (b, c and d in Figure 6.6) are all valid summarized representations of the original data. The behavior of M-cell formulation becomes even more complicated when these M-cells are used to predict test data (data instances with no class labels). These M-cells occupy a different feature space and the arbitrary presence of these M-cells can greatly affect the class association of unseen data. This problem becomes more interesting when data has increasing dimensions and as a result class members are overlapping in the feature space.

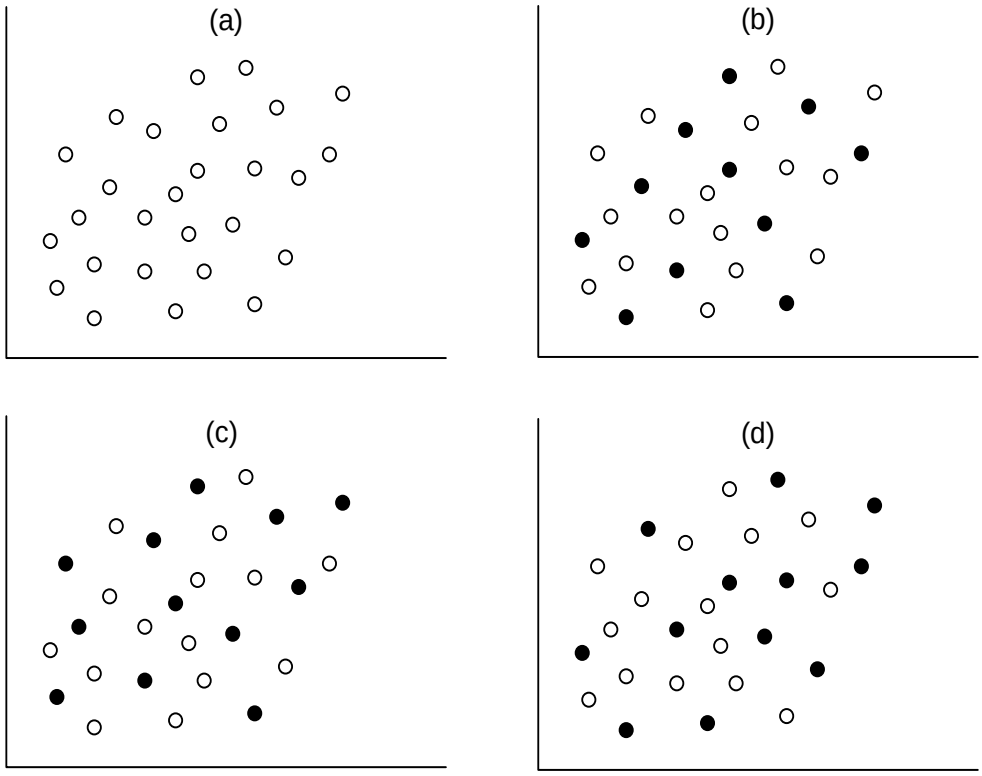


Figure 6.6: Ag presentation order and function of M-cells: (a) simulated data, (b) first set of M-cells (marked), (c) second set of M-cells (marked), (d) third set of M-cells (marked)

6.7.2 Affinity Maturation

One main difference between the unsupervised version of HAIS and the supervised one is the presence of an affinity maturation phase in the supervised HAIS algorithm. In unsupervised HAIS, once an Ag is recognized by Abs or M-cells, it is returned to the appropriate B-cell and the B-cell receptor adapts to the captured Ag completely. This means that the B-cells produce further Abs based on the captured Ag, not on the stimulated Ab that has captured an Ag. In other words, in unsupervised HAIS a perfect 100% learning of Ab from Ag is performed and this standpoint removes the need of fine-tuning or adapting B-cell receptors towards Ag receptors to generate a further population of Abs. Here, in the HAIS supervised learning algorithm, once an Ag is recognized by an Ab the Ab adapts or fine-tunes its receptor in the direction of captured Ag, using a random generation of Abs via a local mutation operator.

Two different sets of experiments were performed in order to observe the effects of using affinity maturation processes (as used in the HAIS supervised learning algorithm) in comparison to 100% learning (as performed in HAIS unsupervised algorithm). In the first set of experiments, the original HAIS supervised learning algorithm as explained in section 6.4 was used, with a β value of 10 (see Equation 4). In the second set of experiments, once an Ab captures an Ag, the Ab completely fine-tunes its receptors towards the Ag (e.g. Ab := Ag). This removes the need of the affinity maturation process in the original HAIS supervised algorithm in steps 9-b-ii and 9-c-i. Each set of experiments was repeated 50 times and the results are shown in Figures 6.7 and 6.8. The results are shown using 1-NN, 3-NN and 5-NN. Figure 6.7 shows the results when a stimulated Ab 100% adapts to Ag receptors while Figure 6.8 shows the results when a β value of 10 is used for the calculation of the AMT parameter (Equation 4) for the affinity maturation of the stimulated Abs. One simple observation to be made from Figures 6.7 and 6.8 is that 3-NN performs better than 1-NN or 5-NN.

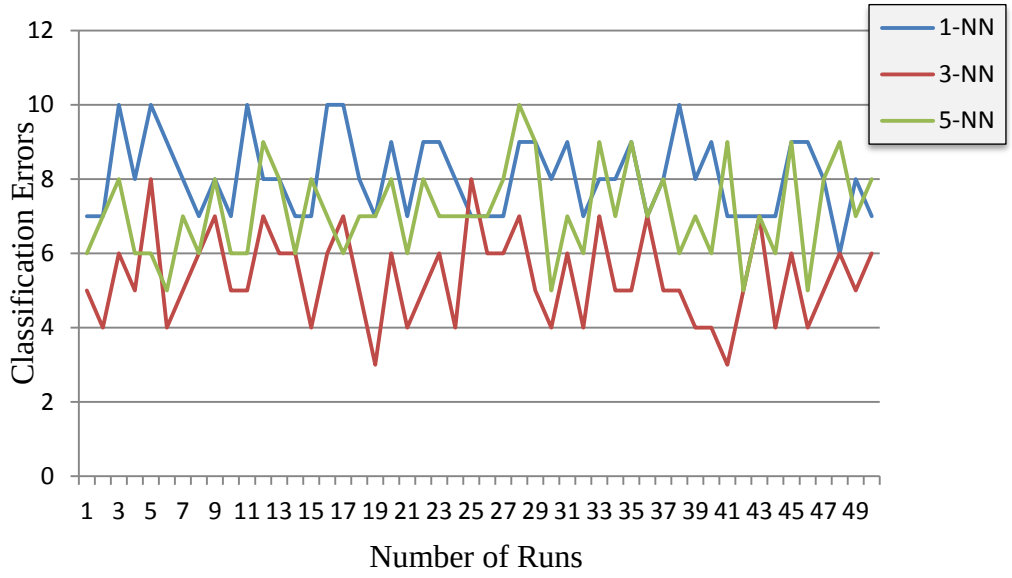


Figure 6.7: 100% adaptation of Ag receptors by stimulated Abs. Results of 50 runs using 1-, 3- and 5-NN are shown.

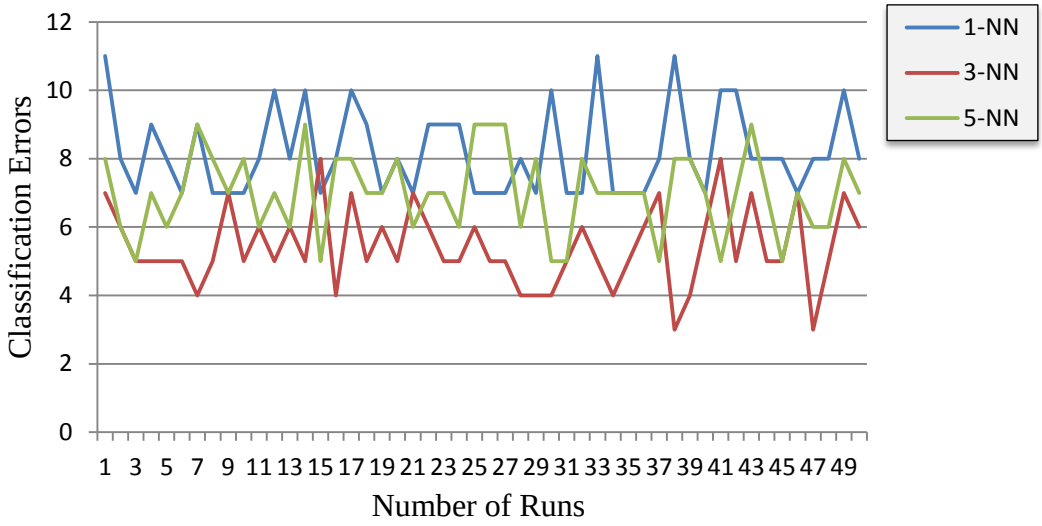


Figure 6.8: An β of 10 is used for affinity maturation to adapt Ag receptors by stimulated Abs. Results of 50 runs using 1-, 3- and 5-NN are shown.

Table 6.7 summarizes the results obtained in Figures 6.7 and 6.8. The mean, standard deviation, minimum and maximum classification errors obtained are given and Table 6.7 demonstrates that using the extra steps of affinity maturation (9-b-ii and 9-c-i) does not add any additional benefits to the performance of the algorithms. The performance of the algorithm 'without affinity maturation' is as good as with the algorithm 'with affinity maturation'. This experiment eliminates the need to use the standard affinity maturation process in the HAIS algorithm.

Table 6.7: Summary of Figures 6.7 and 6.8

	Ab := Ag			β = 10		
	1-NN	**3-NN**	**5-NN**	**1-NN**	**3-NN**	**5-NN**
Mean	8.06	5.36	7.12	8.20	5.42	7
Std.	1.078	1.2081	1.2395	1.2617	1.1622	1.2122
Min.	6	3	5	7	3	5
Max.	10	8	10	11	8	9

6.7.3 The Role of the α and β Parameters

There are two main parameters used in the HAIS supervised learning algorithm, namely α and β; the former controls the stimulation level of Abs/M-cells during interaction with Ags and the latter is known as the degree of learning of stimulated Abs from Ags. We used the same 5-fold Iris data configuration, as explained in section 6.7, to understand the influence and role of various α and β parameters on the final classification results obtained. The results are shown using 1-NN, 3-NN and 5-NN. The first value in each cell of Tables 6.8 and 6.9 is an average classification error obtained over 50 runs and is followed by the standard deviation in parentheses.

We have already shown in Figure 6.5 that the number of M-cells obtained by the algorithm is directly influenced by the α parameter. The value of α and the final number of M-cells have a positive linear relationship. When the α parameter value is set to be one or two, the average lowest classification error of 12.0 was obtained using 1-NN, as there were not many M-cells; also those M-cells are very distant from each other and adding the majority voting system (3- or 5-NN) can seriously affect the class association of unlabeled instances. Therefore, 1-NN was able to produce better classification results than 3- or 5-NN. However, for α values of 3 or 4, there was around 40% to 50% data summarization and 3-NN achieved a better classification result than 1- or 5-NN. The trend continued; for an α value of 6 to 10, 5-NN gave superior classification results. This is because, there are many more final sets of M-cells obtained at the end of a training phase, so more accurate class labels can be assigned using a majority vote from more than 3 M-cells; in this case 5 M-cells are sufficient. The reason for using an α value of 4 earlier in the experiments is that at this value we obtained a minimum classification error of 3 (see Figures 6.7 and 6.8). Furthermore, the task of HAIS is not only to find a better classification but also to achieve a higher degree of data

summarization. Therefore we selected an α value of 4, where errors are relatively lower and data reduction capabilities are good (approximately 50%).

The β parameter is also very important in HAIS, as it controls the degree of learning: how much a stimulated Ab can learn from a captured Ag. The learning values are acquired by using Equation 4. The experiment was run 50 times each using different β parameter values ranging from 1 to 8 and the results are shown in Table 6.9. It can be seen that the β value with 3-NN produced better classification results throughout. Another thing to learn from this experiment is that the average classification error reduces as the β parameter value increases. Here we have only shown experiments with β values of 1 to 8. Earlier in Figures 6.7 and 6.8 we showed that a β value of 10 gives almost same results as a 100% transfer of information from Ag to Ab. Therefore, we can conclude from this experiment that a reasonably high learning ratio of Abs from Ags is required to achieve improved classification accuracy.

Table 6.8: Average classification errors obtained on various α parameter values

	Alpha (α)						
	1	**2**	**3**	**4**	**6**	**8**	**10**
1-NN	12.0 (0)	12.34 (2.370)	9.90 (1.474)	8.60 (1.229)	7.180 (0.388)	7.0 (0)	7.0 (0)
3-NN	38.48 (0.505)	10.32 (2.142)	7.880 (1.944)	5.560 (1.0721)	5.520 (0.505)	6.0 (0)	6.0 (0)
5-NN	59.440 (0.501)	10.36 (1.925)	7.90 (1.764)	7.560 (1.417)	5.380 (0.567)	4.0 (0)	4.0 (0)

Table 6.9: Average classification errors obtained on various β parameter values

	Beta (β)					
	1	**2**	**3**	**4**	**6**	**8**
1-NN	8.140 (1.841)	8.240 (1.721)	8.260 (1.536)	8.240 (1.422)	7.940 (1.420)	8.040 (1.538)
3-NN	6.340 (1.560)	6.480 (1.529)	5.920 (1.243)	5.920 (1.085)	5.820 (1.3702)	5.720 (1.196)
5-NN	7.360 (1.467)	7.240 (1.623)	7.000 (1.294)	7.140 (1.726)	7.080 (1.209)	7.400 (1.370)

6.8 Discussion

An interesting question to be asked here is: Why should we want to construct a supervised AIS algorithm, given that it is not biologically plausible to assume that viruses and other pathogens come with labels that clearly identify them as dangerous or non-dangerous? First of all, we don't *really* know how an NIS works. Initially immunologists believed that the immune system differentiates between self and non-self (external labels). Recently, there has been another theory gaining a lot of attention: 'danger theory' (discussed in previous chapters). According to this theory, the immune system recognizes invaders based on an internal immune response generated in the form of danger and non-danger signals. Secondly, and more importantly, we do not yet know much about the kind of learning performed in an NIS: is it supervised learning or unsupervised learning? It cannot be purely unsupervised, as the system has to be validated with some external objective function (the ultimate test of our immune system is whether we live or die). On the other hand, it cannot be purely supervised, as pathogens clearly do not come with labels (self, non-self) that can guide an immune system to trigger an appropriate immune response. Therefore, we strongly believe that the immune system performs semi-supervised learning. In semi-supervised learning the learning is performed using a hybrid of both supervised and unsupervised learning techniques, as data consists of both labeled and unlabeled instances (examples).

Supervised and unsupervised learning algorithms are two ends of a very wide spectrum of machine learning techniques. Semi-supervised learning covers the rest of the spectrum. In this research, we have investigated AIS-based unsupervised learning approaches in previous chapters and now we have proposed a supervised AIS learning approach to cover the other end of the spectrum. At present, we are not completely aware of the degree of semi-supervised learning performed in an NIS. Therefore, in this research we are trying to cover both extremes of the spectrum.

6.9 Summary

The focus of this chapter was to develop a novel supervised learning algorithm inspired by NISs. Watkins [88] showed with experimental results that AIRS is more than comparable with other traditional supervised learning techniques, such as neural networks, k-NN and C4.5. This chapter has shown that our proposed HAIS algorithm

performs as well as AIRS and on some datasets actually outperforms it. Some of the important concepts of an NIS, such as B-cells, M-cells, negative clonal selection and affinity maturation were used to build our AIS and we argue that this adds a greater degree of biological plausibility to the HAIS. A more active role for M-cells is proposed and used in the HAIS. Negative clonal selection, which is replaced by ARBs in AIRS, is also used in our algorithm. Two separate affinity threshold measures, namely global AT and local AT were introduced in order to avoid over-fitting, and the experimental results were presented to demonstrate its effectiveness. In particular, the local AT that we introduced can help to deal with datasets that have non-homogeneous class structures.

The general behavior of the parameters of the HAIS supervised learning algorithm was investigated and the results reported in this chapter. The arbitrary formulation of M-cells due to the random Ag presentation order was also examined. The role of M-cells in achieving better classification accuracy was also investigated and reported. We found that various Ag presentation orders can produce different classification results. We also extensively discussed the roles of affinity maturation process and M-cells and concluded from our experimental results that it is the spatial positioning of M-cells which contributes most profoundly towards achieving low classification error on unseen data.

The final M-cells obtained by the HAIS algorithm and k-NN method were then used to validate the HAIS algorithm. It was also demonstrated with experimental results that the value of k in k-NN is dependent on the α parameter value selected. A more robust validation method must be considered that can also take care of outliers in the testing/validation data. This may be achievable by using a mutation strategy that is more sophisticated in the final M-cells.

In summary, the chapter reported on novel work that is situated at a very early stage of our understanding of the NIS and of the advantages that can accrue through adding greater biological plausibility to AIS algorithms as our understanding of micro-level processes in NISs grows. This chapter has opened up more avenues of research than it has closed down, which bodes well for future AIS research.

AIS algorithms including HAIS are known to possess data reduction capabilities through M-cells. However, there is currently very little understanding of how effective M-cells are at reducing the data (i.e. for generalizing to important structural properties

of the data) while at the same time preserving critical class information. Furthermore, it would also be interesting to see whether the introduction of M-cells aids or hinders the artificial learning process at a suitable learning stage. Vaccination is another very important learning process in NIS which has not been discussed or explored so far. Vaccination is the process of stimulating the immune system by using a weaker infectious agent or extracting proteins from an infectious agent. The immune system can mount a response against the partial virus or bacteria. This process provides immunity from even more harmful pathogens of the same kind in the future and the body becomes immune to such pathogens for almost the rest of its life. In the next chapter, the role of Abs/M-cells will be discussed in the context of vaccination of an artificial learning system.

Chapter 7

Vaccination of Learning Systems: Principles and Methods of AIS

Once the body gains immunity to a specific disease, it generally remains free from it almost for life. One way to build such immunity is through vaccination. Vaccination is the process of stimulating the immune system by using an infectious agent in a weaker form or proteins extracted from an infectious agent [65]. A vaccine typically activates an immune response by generating antibodies, which are cloned and hypermutated to bind to antigens (fragments) of pathogens. The main aim of this chapter is to explore the effectiveness of the artificial vaccination of learning systems by injecting vaccination material into the learning system to improve its learning capabilities.

In previous chapters, we have introduced the Humoral-Mediated Artificial Immune System (HAIS) supervised learning algorithm, which is inspired by the way an adaptive immune system can recognize and trigger an immune response through generations of antibodies. The HAIS supervised learning algorithm, like other artificial immune system (AIS) algorithms, produces memory cells which are used to classify unseen data patterns. Memory cells are regarded as a summarized form of the original data. Therefore, the final set of memory cells can also be seen as a form of vaccine (proteins

extracted from pathogens) that can be used in a learning system to enhance its learning capabilities. Artificial neural networks (ANNs) are used to model the learning process together with the HAIS to synthesize the vaccination material for injecting into the learning process. Put simply, the primary objective of this chapter is to demonstrate that effective and improved learning can be achieved by applying the concept of vaccination. Two other interesting phenomena of natural immune systems (NISs) are immunosuppression and autoimmune diseases. These phenomena are also explored and discussed in terms of the hypermutation of antibodies.

The rest of this chapter is structured as follows. Section 7.1 explains the motivation of the chapter and highlights the importance of a vaccination process in building a supervised learning model. Section 7.2 explains the proposed methodology which involves a hybrid architecture consisting of the HAIS and an ANN. Section 7.3 demonstrates the feasibility of the proposed methodology with experimental results conducted on simulated and real-world datasets. Finally, a summary of the chapter is presented in section 7.4.

7.1 Introduction

The immune system can deal with previously unseen pathogens and viruses based on existing knowledge and can also trigger much faster responses to already seen pathogens (if in the future same pathogen attacks again) because of the immune memory attained by an NIS during the previous attack. Therefore, once an NIS is exposed to a disease and new pathogens, it adapts and remains immune for long periods after the disease has been combated. One very successful method of learning in the adaptive immune system is vaccination, where inactive or weakened forms of the pathogens are introduced into the NIS to stimulate a defensive response without leading to the disease [65]. Vaccines lead to the production of antibodies so that the NIS is primed should the strong version of the pathogen be encountered in the future. Vaccines are categorized by composition and formulation (how they are derived, how they are used, and how their effects are mediated). For example, the tetanus vaccine is produced from the toxic chemicals (antigens) extracted from the tetanus pathogen, as are the vaccines for hepatitis B and diphtheria [65].

Two of the main components of an adaptive immune system are memory cells and antibodies, which help the NIS fight pathogens and subsequently eradicate them (with the help of other NIS components). Memory cells encourage specialization by keeping track of already encountered pathogens and, if the same pathogen attacks the body in the future, the presence of memory cells can trigger a faster response to eliminate these pathogens by producing the relevant antibodies. Once released by memory cells, antibodies circulate in the body and help recognize and eliminate similar pathogens by binding to pathogenic antigens. Even as memory cells proliferate to produce antibodies, the paratope of the antibodies (that is, the sequence of amino acids that binds to the epitope on an antigen) can mutate with high frequency so that they are able to bind better with the antigens. This mutation, called hypermutation, is commonly used in AIS clustering/classification approaches to find improved solutions on the assumption that pathogens/antigens are data samples and antibodies and their cells are classes or clusters. But despite the importance of hypermutation in AIS, there is relatively little understanding of its behavior or of how to use it most effectively in learning. This chapter explores immunosuppression and autoimmune disease in the context of hypermutation within AIS. Immunosuppression refers to reduced activation of the immune response, whereas an autoimmune disease arises from an over-activated immune response of the body [65]. One of the most significant potential benefits of using memory cells in AIS is their data reduction/generalization capability. The objective of data reduction is to simplify the representation of the data while keeping the core information of the original data intact. This chapter will demonstrate that memory cells obtained using the HAIS algorithm can also provide effective data reduction capabilities.

Hart and Timmis [10] proposed that AISs be incorporated with other biologically inspired techniques such as neural networks, swarm algorithms and genetic algorithms to realize their full potential. The aim of this chapter is to bring together two existing strands of research, namely AIS and ANNs to form a new hybrid architecture, where the AIS is embedded in an ANN to help the ANN learn through memory cells and hypermutated antibodies. Learning capabilities, highly specialized cells, diversity and memory are some of the common features of both systems [64]. ANNs are inspired by the structural and functional aspects of the biological neural networks in the human

brain. They are very well established in the fields of classification and machine learning in general [14, 20].

The task of any classifier (in supervised learning) is not only to build up a model that can produce minimum classification errors on training data, it is also to generate a model that can classify unseen and future patterns efficiently and accurately, i.e. a model that generalizes to unseen data. Typically, a k-fold technique is used to assess the generalization capabilities of a classification algorithm. The main objective of this chapter is to demonstrate that vaccination of an ANN through memory cells and antibodies does not reduce the ANN's learning capability and in some cases can improve on it. Our hypotheses in this chapter are as follows:

H1: *Training an ANN with memory cells prior to exposure to the original data leads to improved and effective learning (i.e. memory cells as data reduction are effective).*

H2: *Effective learning of unseen pathogens and the classes to which they belong can be achieved by vaccination of learning systems.*

7.2 Proposed Methodology

The main objective of the proposed methodology in this section is to investigate whether the introduction of memory cells at a suitable stage of learning (a form of vaccination) aids or hinders the learning process in an ANN. Memory cells are considered to be a subset of the original data or, in this case, proteins extracted from pathogens (referred to as acellular vaccines). The HAIS algorithm proposed for supervised learning is used to extract memory cells (vaccine) from the data which are then used to generate antibodies with different mutation rates, which can be seen as various levels of activated immune response. In this chapter, the mutation rate of antibodies varies from 5% to 40% depending on the dataset used. Negative clonal selection (a salient feature of the HAIS algorithm) is performed on generated antibodies to maintain diversity and to avoid over-population. The ANN is used to model the effectiveness of the memory cells/antibodies.

The ANN used here can be considered to be a virtual learning organism that may be experimented on, and the HAIS as a subpart of the ANN to produce the vaccination material. A simple back propagation and logistic activation function-based ANN was

used for the experiments. JavaNNS (a version of SNNS) [192] was used to implement the ANN. Further details regarding ANN architecture, back propagation and various activation functions can be found in [41]. The vaccination experiment proposed here has two stages:

A. There is no prior immune system in the ANN.
B. There is a prior immune system in the ANN.

Stage A is achieved by using all the data to train the ANN after exposure to the vaccine (there is no test phase), and Stage B by separating the full data into training and test data and then initially training the ANN on the training data only prior to checking the generalization capabilities on the test data. Therefore, vaccines (memory cells) are extracted either from the whole dataset (in the case of Stage A) or only from test data (in the case of Stage B). The complete methodology can be explained in three phases which are shown in Figure 7.1. It is necessary to prove three main conditions to validate our proposed hypotheses:

a. The effectiveness of memory cells in learning systems.
b. The effectiveness of antibodies obtained from memory cells in learning systems, where no prior immune system exists.
c. The effectiveness of both memory cells and/or antibodies when a prior immune system exists.

Therefore, we devised three sets of experiments (or conditions) to systematically test the effectiveness of our proposed methodology. These sets of experiments or conditions are explained in Figure 7.1.

7.2.1 Condition 1 - C1

It is assumed that no prior immune system exists in the ANN learning system. All the data is assumed to be a pathogen attack. The vaccination is produced in the form of memory cells from the data using the HAIS supervised learning algorithm. The obtained vaccine (memory cells) is injected into the learning system (ANN) before the ANN is exposed to full pathogen attack (the complete data). In simple terms, memory cells are used to train the ANN and once learning is completed (convergence is achieved), the whole dataset is then used to further train the ANN. This process is shown in Figure 7.2.

Phase I (P-I) of the proposed methodology is missing as we are assuming that no prior immune system exists.

7.2.2 Condition 2 - C2

As explained earlier, once a vaccine is injected into the human body (learning system), it triggers an immune response by generating antibodies using hypermutation. In C1, a vaccine (memory cells generated by the HAIS algorithm) is introduced into the system and the effectiveness of that vaccine is observed. Here, in C2, separate population sets of antibodies are generated from memory cells using different mutation rates. These generated antibodies are used to train the ANN (separately), prior to training the ANN on the original data to observe the effectiveness of a vaccination process in the learning systems. This process is shown in Figure 7.3.

7.2.3 Condition 3 - C3

Here we consider the circumstance where a prior immune system exists. To demonstrate this we have divided the original data into a training set and test data. Memory cells and antibodies are generated from the test data. Now, in the ANN learning system, initial learning is performed on the training set (P-I). In the second phase (P-II), memory cells or antibodies along with the training data are used to train the ANN. Finally, the test data is used to again train the ANN. The entire process is shown in Figure 7.4.

Initial phase (P-I): Train ANN on Training data if primary immune system exists, or skip this phase

Vaccination phase (P-II): Use vaccination (memory cells or antibodies) as well as training data to train the ANN.

Pathogen exposition phase (P-III): Once the ANN has converged during the vaccination phase, expose original pathogens (or testing data) to the ANN.

Figure 7.1: The three principal phases of the proposed methodology

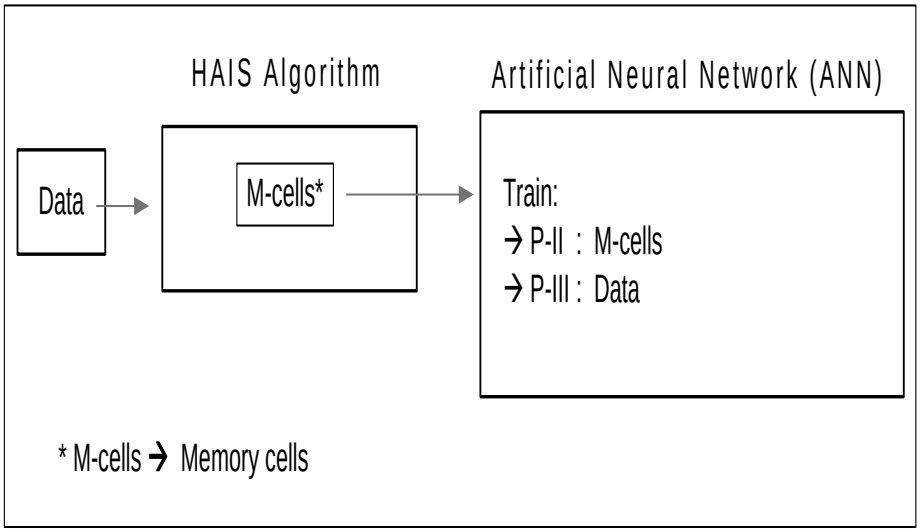


Figure 7.2: C1: Memory cells are obtained from the data using the HAIS algorithm. The ANN is primarily trained (until convergence is achieved) on memory cells. Once the model converges, (the original) data is introduced and training of the model is performed again until convergence

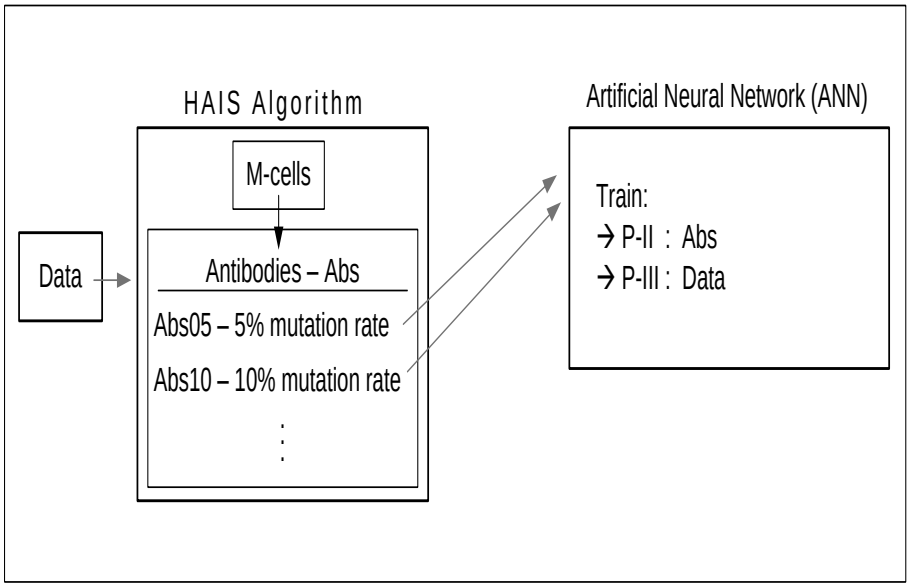


Figure 7.3: C2: Memory cells are obtained using the HAIS algorithm from the data. Antibodies with different mutation rates are generated and stored separately. ANN models are generated using different mutation rates of antibodies prior to introduction of (the original) data.

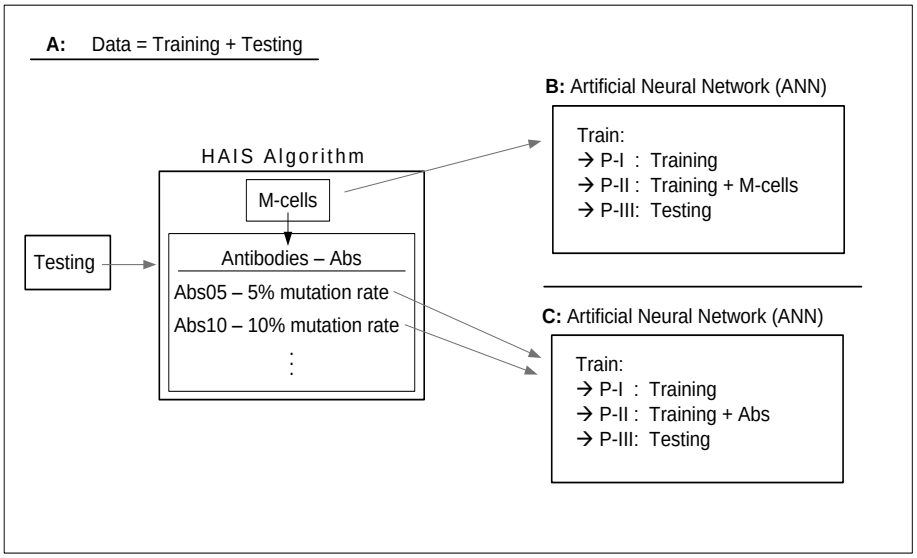


Figure 7.4: C3: Vaccination of learning systems where a prior immune system exists. (A): Data is split into training and test data. Only the test data is used to obtain memory cells, and then antibodies using different mutation rates are generated separately. (B): The ANN model is trained using training data and, after convergence, training + memory cells are used to train it again until convergence. Then the test data is used for training the ANN model. (C): Here, the same process as in (B) is performed but antibodies are used instead of memory cells to train the ANN.

7.3 Experimental Results

One simulated and four well-known real-world datasets, namely Iris, Breast Cancer Wisconsin, Parkinson's Disease and Statlog (Heart) (see appendix A) were used. All four of the real-world datasets are available from the machine learning repository at UCI [193]. The simulated dataset was used to test whether the proposed method can find perfect classification when both classes are linearly separable. The main purpose of this chapter is not to compare our results against existing state-of-the-art techniques, but to demonstrate that memory cells and the information they contain can be exported to other techniques (in this case, ANNs) to produce models that are more robust and generalized than using those techniques alone. The ANN architecture for the simulated dataset was 2x(6x10)x1 (two input nodes, a hidden layer of 60 nodes configured 6 by 10, and one output node); for the Iris dataset 4x(3x10)x3; Breast Cancer Wisconsin dataset 30x(4x15)x1; for the Parkinson's Disease dataset 22x(3x15)x1; and for the Heart dataset 13x[(1x15), (1x15)]x1. The Statlog (Heart) dataset has two hidden layers of 15 nodes each. The following experiments were conducted in the order of the conditions specified above, namely C1, C2 and C3.

7.3.1 Experiments with C1

The Iris data was used for the following experiment to demonstrate C1. The HAIS supervised learning algorithm produced 66 memory cells out of 150 data instances (56% data reduction). The α and β parameters were set 4 and 5 respectively. The ANN converged to 2 errors (error sum of square [ESS]) after about 7,000 cycles when the original and full Iris dataset (i.e. without memory cells) was used to train the network, as can be seen in Figure 7.5 (Black). However, in another experiment, when memory cells were used to train the ANN it converged to zero error after about 25,000 cycles. Later, when original and full Iris dataset was introduced the error curve converged at 2 errors again (Figure 7.5 [Blue]). The ANN in both cases converged to exactly the same error, which indicates that data reduction in the form of memory cells did not affect the ANN. The results indicate that memory cells possess excellent data summarization/reduction capabilities for fitting the data. Memory cells prime the ANN in such an efficient manner that when the real data (pathogens) attack the ANN, it is well prepared to handle those pathogens and takes comparatively fewer iterations than the original data to converge.

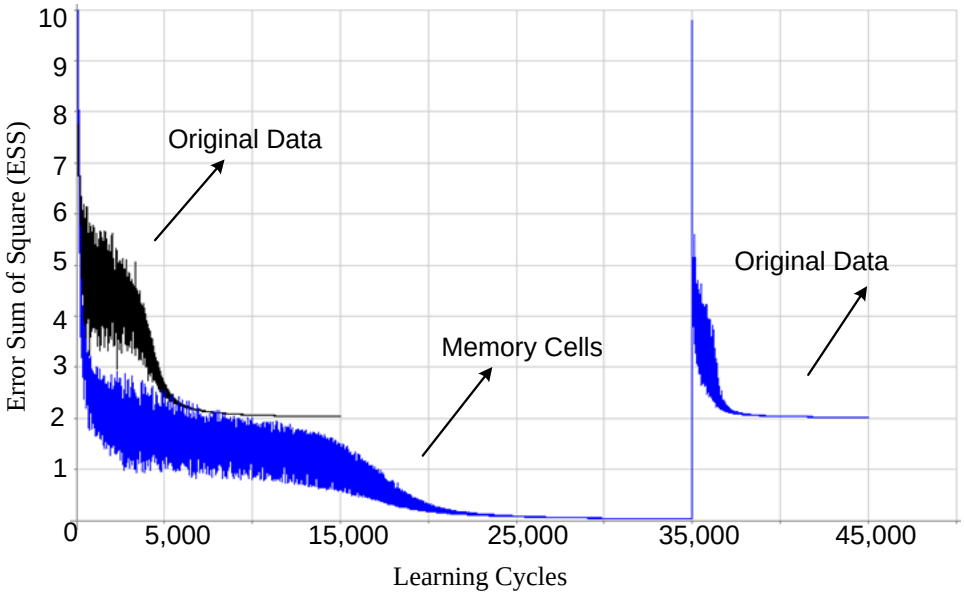


Figure 7.5: Learning error curves of both the original data and memory cells + original data. The x-axis represents learning cycles (iterations) and the y-axis the error sum of square (ESS) obtained for the Iris data. This experiment demonstrates that there is no information loss while generating memory cells, as both (Black and Blue) experiments converge to the same ESS.

In the previous chapter, we demonstrated the effects of the HAIS parameters, namely, α and β, on classification accuracy using real-world datasets. Here, we are examining the effects of the same parameters on achieving effective learning in our hybrid architecture. The β parameter is the degree of learning of the HAIS algorithm when antibodies capture new pathogens. In the previous chapter, we empirically showed that a high degree of learning (by antibodies from antigens) is necessary for effective classification results. Here, experiments were conducted using the C1 configuration. Ten independent runs were performed to obtain memory cells for different β values and constant α value of 4. These memory cells were later used to vaccinate the ANN. The results from using β values of 5%, 10% and 100% learning are shown in Table 7.1. Again, it was found that better results (in terms of effective learning) were obtained using a β value of 10 and using a 100% learning rate.

Table 7.1: Learning errors (ESS) obtained by using various β parameters. Indexes 1-10 refer to the 10 separate runs

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Average**
β = 5	2	0	2	2	2	0	2	2	2	2	1.6
β = 10	0	0	0	0	0	2	2	2	2	0	0.8
Ab:=Ag (100%)	0	2	0	0	0	0	2	2	0	0	0.6

The other HAIS parameter is α, which controls the stimulation level of antigen/antibody interaction. In the previous chapter, the effects of the α parameter on classification accuracy were investigated and it was shown empirically that the α value play an important role in achieving good classification results. A too low or too high α value reduced classification accuracy. The effect of the α parameter was again investigated on the effectiveness of learning systems here. The results are shown in Table 7.2, where 10 independent runs were performed using a 100% learning rate for β and for various α values. It can be seen from Table 7.2 that the performance when the α value is 4 is superior to others on average. Observe that for ‘very low’ or ‘very high’ values of α (in this case α of 2 or 5), the learning system consistently produced 2 errors from the data.

Table 7.2: Learning errors (ESS) obtained by using various α parameters. Indexes 1-10 refer to the 10 separate runs

Index	1	2	3	4	5	6	7	8	9	10	Average
$\alpha = 2$	2	2	2	2	2	2	2	2	2	2	2
$\alpha = 3$	0	2	2	2	2	2	2	2	2	2	1.8
$\alpha = 4$	0	2	0	0	0	0	2	2	0	0	0.6
$\alpha = 5$	2	2	2	2	2	2	2	2	2	2	2

The general trend of using different α parameters for C1 can be seen in Figure 7.6. It is interesting to see that for α values of 2 and 4, when memory cells were used to train ANN they converged to zero error. The convergence of $\alpha = 2$ was more rapid than that of $\alpha = 4$. However, when the original Iris data was introduced, the ANN learning system with an α value of 2 took a much longer time to converge than with an α value of 4. In fact, an α value of 4 took only a few iterations to converge. In the case of α of 5, there were around 95 memory cells produced by HAIS. The ANN using memory cells took almost the same number of iterations to converge as the original data (to 2 learning errors) and once the original data is introduced, it takes a few iterations to converge to 2 errors. When the memory cells were obtained using an α value of 2 and the original data was introduced, the ANN took a longer time to learn (converge to) from the patterns, which suggests there were inadequate memory cells to generalize the original data. But when α was set to 5, the ANN took no time to converge. These experiments suggest that the α and β parameters play a very important role in achieving effective learning.

In the next experiment the Parkinson's Disease dataset was used. The dataset produced zero classification errors using both the ANN on the complete dataset and C1. To demonstrate the effectiveness of our proposed method, we randomly divided the whole dataset (195 instances) into 45 training data and 150 test data. Here, we run C1 on test data. When only the test data was run, the ANN converged to 6 classification errors. When memory cells were used to train the ANN prior to introducing the original test data, the ANN converged to zero classification/learning errors (ESS). The results can be seen in Figure 7.7, where a learning curve on the original test data is shown in black and C1 in blue.

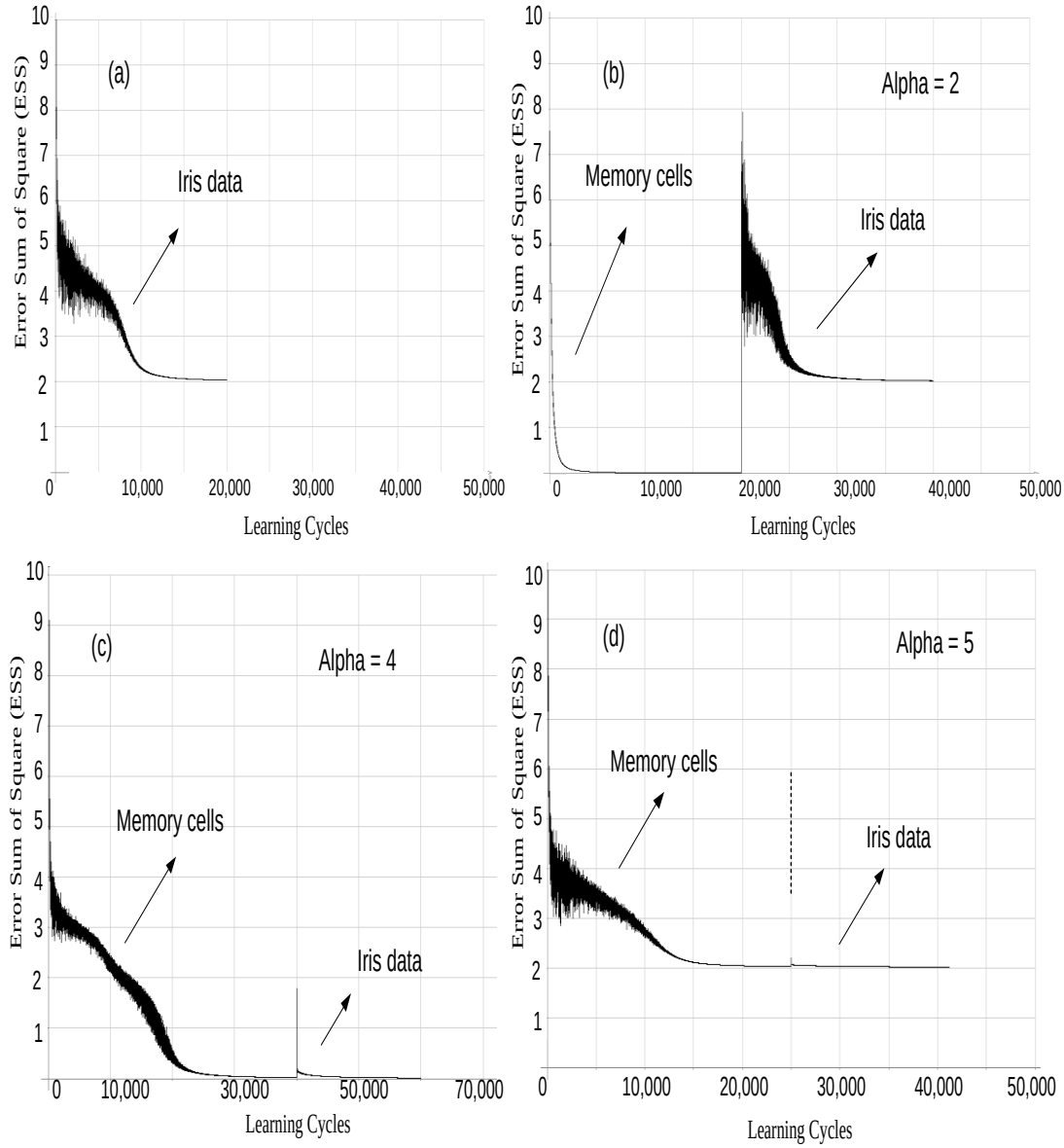


Figure 7.6: Learning error curves of the Iris data with α values of 2, 3, 4 and 5 using C1. (a): When only the Iris data was used to train the ANN, it converged to 2 ESS. (b): Memory cells were obtained using the HAIS supervised algorithm, where the α parameter was set to 2. Initially, the ANN was trained using the memory cells and when the model converged the Iris data was used to train the ANN again. The model finally converged at 2 ESS (with a high degree of learning required when the Iris data was introduced in the ANN). (c): Memory cells were obtained using the α value of 4 and the ANN used to train memory cells before the Iris data was used. The model finally converged at 0 ESS (with very little learning required when the Iris data was introduced in the ANN). (d): Memory cells were obtained using an α of 5 and the ANN used to train memory cells before the Iris data was used to train the ANN again. The model converged at 2 ESS (with minimal learning required when Iris data was introduced in the ANN).

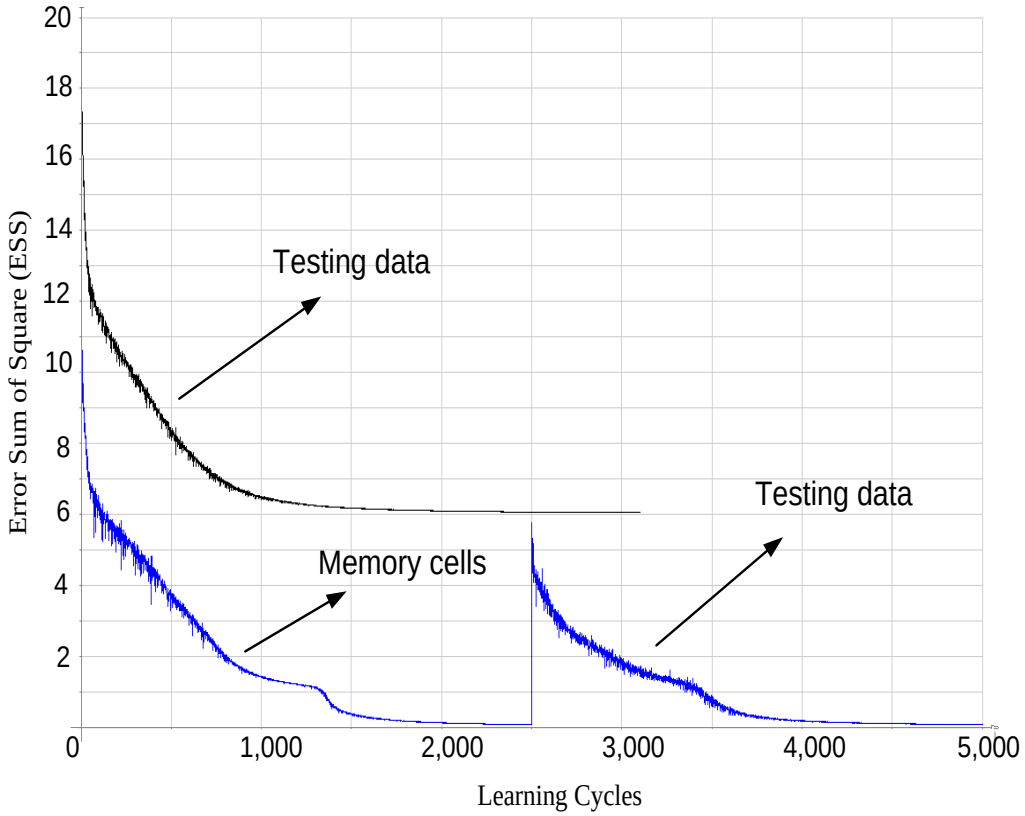


Figure 7.7: Learning error curves of the Parkinson's Disease data, with and without the introduction of memory cells. The ANN using the Parkinson's Disease data (test data only) converged to 6.0 ESS. However, when memory cells were used to train the ANN, it converged to zero ESS and when the Parkinson's Disease data was introduced into the learning model the ANN finally converged to zero ESS.

In the following experiment the Statlog (Heart) dataset is used (for a description of this dataset see appendix A). The ANN using the original data converged to 6 learning errors. On the other hand, when C1 was used, the final learning error dropped down to 2. The learning curves can be seen in Figure 7.8, where the black curve represents the original data and the blue curve C1.

In the next experiment the Breast Cancer Wisconsin dataset was used (for a description of this dataset see appendix A). The learning error curves, using the original data and C1 respectively, can be seen in Figure 7.9, where the ANN using only the original data found 3 classification errors and C1 found 2 classification errors.

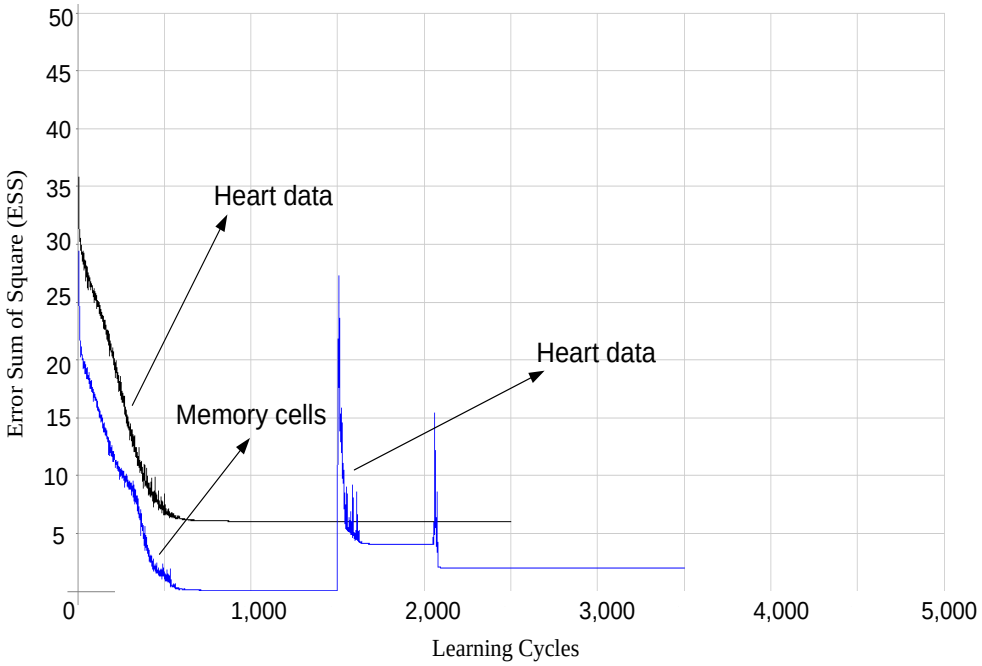


Figure 7.8: Statlog (Heart) learning curve for the original data as well as for C1. The ANN using the Statlog (Heart) data converged to 6.0 ESS. However, when memory cells were used to train the ANN, it converged to zero ESS. When the Statlog (Heart) data was then introduced into the model the ANN finally converged to 2.0 ESS.

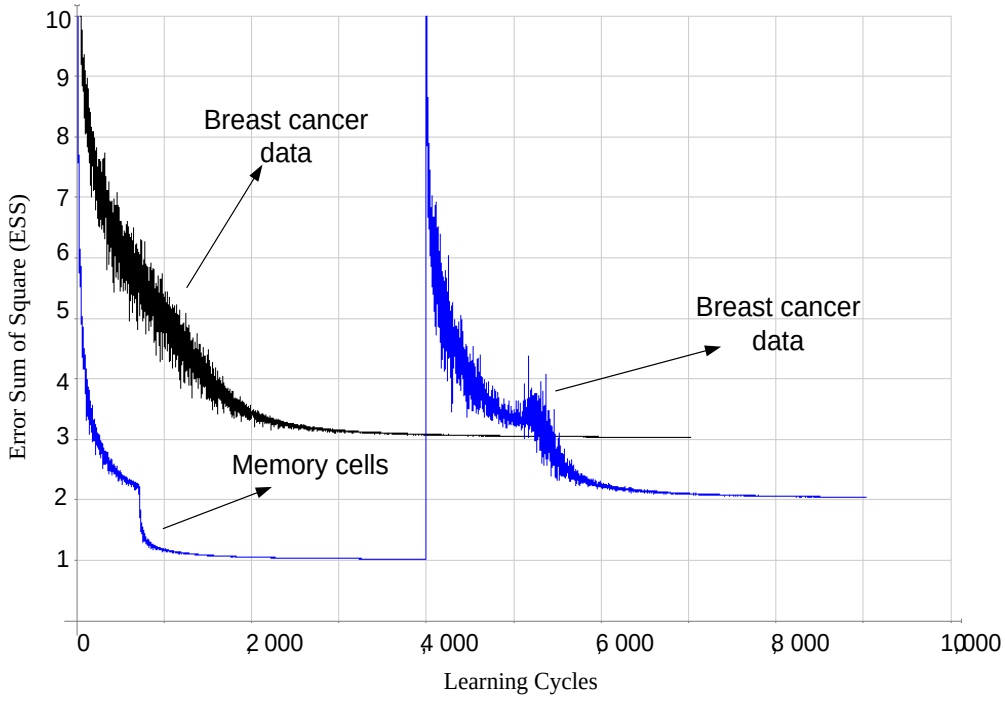


Figure 7.9: Breast Cancer Wisconsin learning curve for the original data as well as for C1. The ANN using Breast Cancer Wisconsin data converged to 3.0 ESS. However, when memory cells were used to train the ANN, it converged to 1.0 ESS and when Breast Cancer Wisconsin data was then introduced into the learning model, the ANN finally converged to 2.0 ESS.

So far we have demonstrated with experimental results that improved and effective learning can be achieved by introducing memory cells in the ANN prior to the original data (C1). The experiments conducted above show that vaccination (memory cells) does help in building an efficient learning model on seen data, when only the vaccine is used to train the ANN prior to exposing it to the original data. However, in the vaccination process, once a vaccine is injected into the body it starts to produce antibodies and this helps to effectively generalize the learning of the immune system on seen and subsequently on unseen data.

7.3.2 Experiments with C2

In the above experiments, we have demonstrated that memory cells (vaccination material) possess excellent learning capabilities. In the following experiments we will try to establish that the vaccination process also helps to improve the performance (in terms of learning capabilities) of the ANN. 'Vaccination process' here means generating antibodies with varying mutation rates from memory cells. Put simply, we are investigating the effects of the introduction of antibodies prior to the original data into learning systems (described above in C2).

To demonstrate the effectiveness of a vaccination process and hence the generation of antibodies, we selected the memory cells that produced 2 final learning errors in C1 using the Iris data (see Table 7.2). These memory cells were used to produce a separate population of antibodies using different mutation rates. For example, a population of antibodies was produced using a maximum mutation rate of 5% and named Abs05; then a population of antibodies was produced using a maximum mutation rate of 10% and named Abs10, and so on. The results obtained from the exposure of the antibodies to the ANN before and after the original Iris data can be seen in Figure 7.10. It can be seen from that a learning system generating antibodies with 10% mutation converged to zero learning errors when the original data was introduced. The results of the 5% mutation rate were no different to those of the unmutated memory cells at the final stage. The final classification results also deteriorated when a 15% mutation rate was used. Another observation here is that as the mutation rate increased, the learning error on the antibodies also increased. It can be argued that this is because we are using random mutation on each feature to generate antibodies and more generalization of data is taking place at the price of higher learning errors on the antibodies (seen data). Here, we

can see a trade-off between the generalization of data and learning error. Perhaps a controlled or specialized mutation function could be helpful in reducing learning errors. In this chapter we do not introduce a sophisticated mutation function as the main objective here is to show that efficient and improved learning can be achieved using a vaccination process.

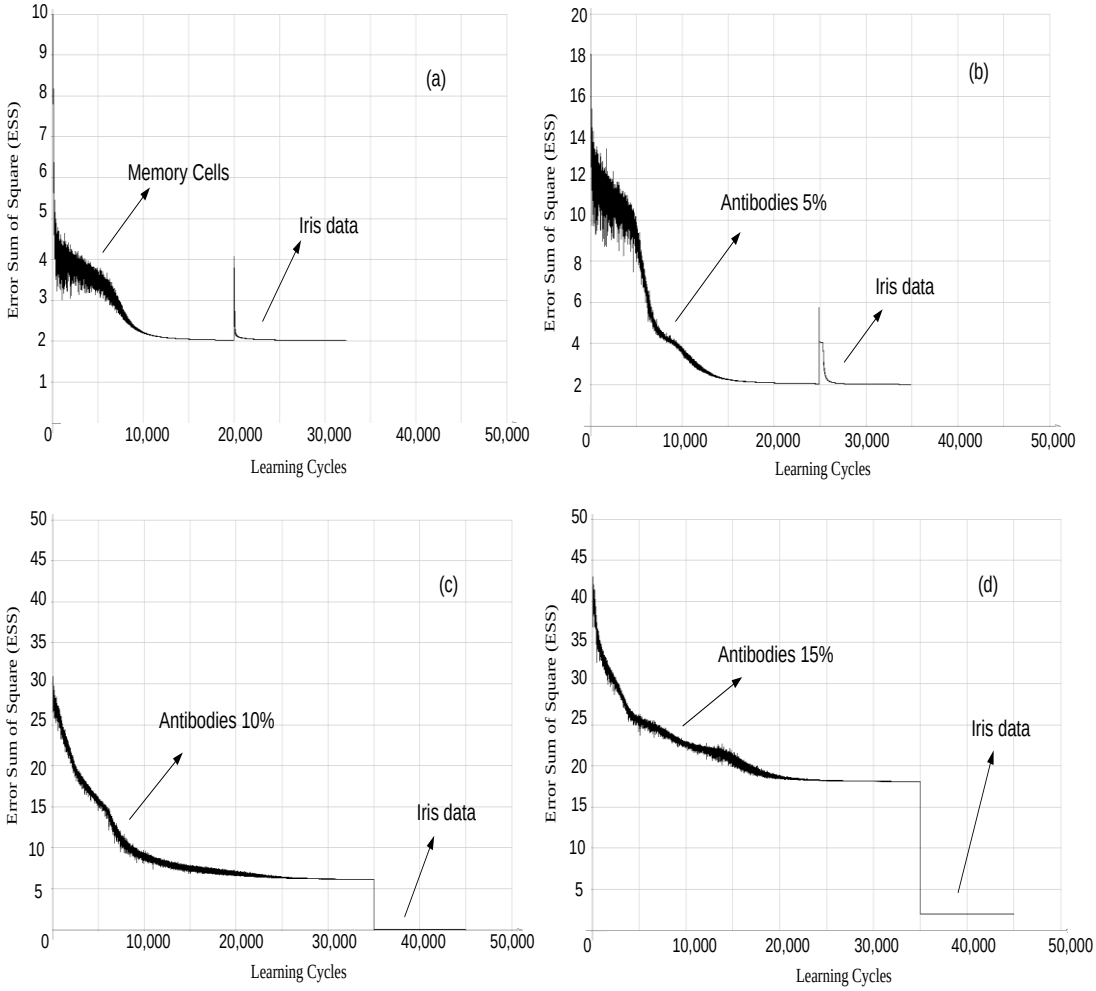


Figure 7.10: C2: Effects of introducing antibodies prior to the original data using the Iris data in an ANN learning system. (a): Memory cells were used to train the ANN before introducing the Iris data and the model finally converged to 2.0 ESS, which is the same as result from the ANN using only the Iris data. (b): When a 5% mutation was used to generate antibodies and the ANN was trained on them before introducing the Iris data, the ANN model converged to the same 2.0 ESS. (c): When a 10% mutation rate was used to generate antibodies and the ANN was trained on those antibodies before introducing the Iris data, the ANN model converged to 0.0 ESS. (d): When a 15% mutation rate was used to train the ANN before introducing the Iris data, the learning errors increased to 2.0 ESS again.

Next a simulated dataset with two distinct classes was used next to further demonstrate the effectiveness of antibodies and hence the C2 method, as well as to verify the

learning results obtained in earlier experiments (Figure 7.10). The simulated data had two features and consisted of 153 instances. The 2-D projection of the simulated data can be seen in Figure 7.11 (R). A total of 34 memory cells were produced using HAIS: 16 were from class 1 and 18 from class 2, resulting in 78% data reduction. The memory cells were introduced into the ANN and the ANN run until convergence was achieved (3,000 iterations). The original data was then introduced into the trained ANN as shown in Figure 7.11 (L). The ANN's errors momentarily went higher and then rapidly converged to zero error ESS. This experiment indicated that the memory cells extracted from HAIS are an accurate representation of the data in reduced form for the purpose of supervised learning (C1). In other words, the structural properties of the data are kept intact in the memory cells.

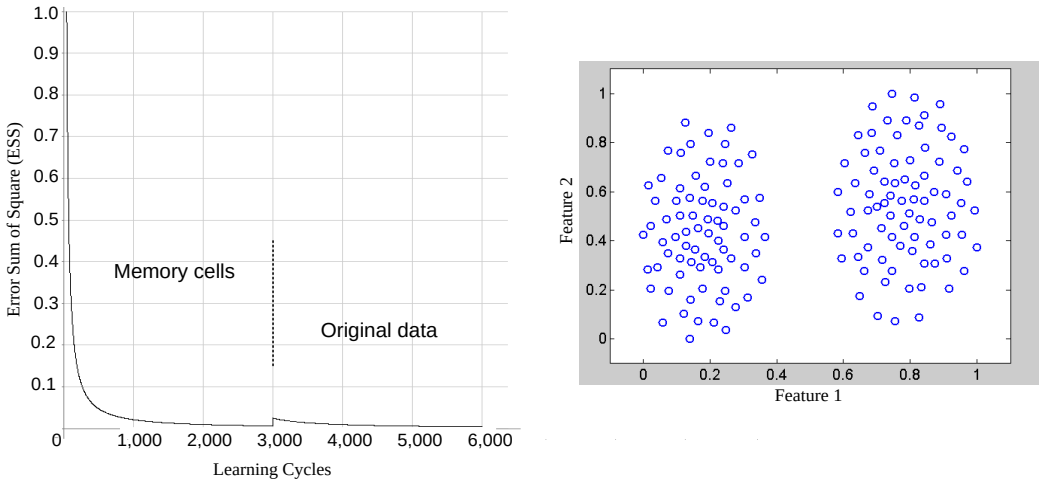


Figure 7.11: L: ESS curve, where the x-axis represents learning cycles (iterations) and the y-axis ESSs obtained for the simulated data. R: The original projection of two-class data.

The main purpose of any vaccine is to trigger production of antibodies (secondary response). The role of this secondary response is evaluated in the following experiments using the same simulated data. The memory cells generated in the above experiment were used to generate antibodies with different mutation rates ranging from 5% to 40% (C2). Separate ANNs were then trained using these antibodies and after convergence the original data was introduced to the trained ANNs. In Figure 7.12 (a) it can be seen that when mutation of 5% and 10% of antibodies was used, the ANN took 5000 iterations to converge. When original data was then introduced, the ANN stayed stabilized and the ESS stayed at zero. The same behavior can be seen in Figure 7.12 (b), where 20% mutation was used to generate antibodies. An interesting behavior started to emerge

when a 25% mutation rate was used to generate antibodies (in (see Figure 7.12 [c]). Once the ANN converged to zero after 320,000 iterations, on exposure to the original data the ANN learning error curve increased and then stabilized at 2 ESS. Exactly the same behavior was observed for a 40% mutation rate (Figure 7.12 [d]). In this case, the final learning error on the original data stabilized at 4 ESS. These experimental results suggest that there was an over-generalization of the antibodies. Also, as the rate of mutation increases, the ANN needed more iterations to converge. This is because the antibodies started to overlap in feature space, which means more time (iterations) was required for convergence to the minimum error. This was in turn due to a reduction in linear separability and over-generalization of data leading to an increased classification/learning error.

As noted at the start of the chapter, immunosuppression refers to under-activation of immune response and an autoimmune disease arises from an over-activation of the immune response. In our experiments, generations of antibodies with different mutation rates can be regarded as activations of immune response at different levels and magnitudes. Lower mutation rates can be considered as under-activations of immune response or immunosuppression, whereas higher mutation rates can be seen as over-activations of immune response or autoimmune disease. In the case of Figure 7.12, antibodies with 5% and 10% mutation rates can be regarded as immunosuppression and antibodies with 25% and 40% mutation rates can be seen as autoimmune disease.

The above sets of experiments were repeated 10 times for each mutation rate and the overall averages are reported in Table 7.3. For antibodies with mutation rates of 5%, 10% and 20%, the ANN always found zero errors on the original data.

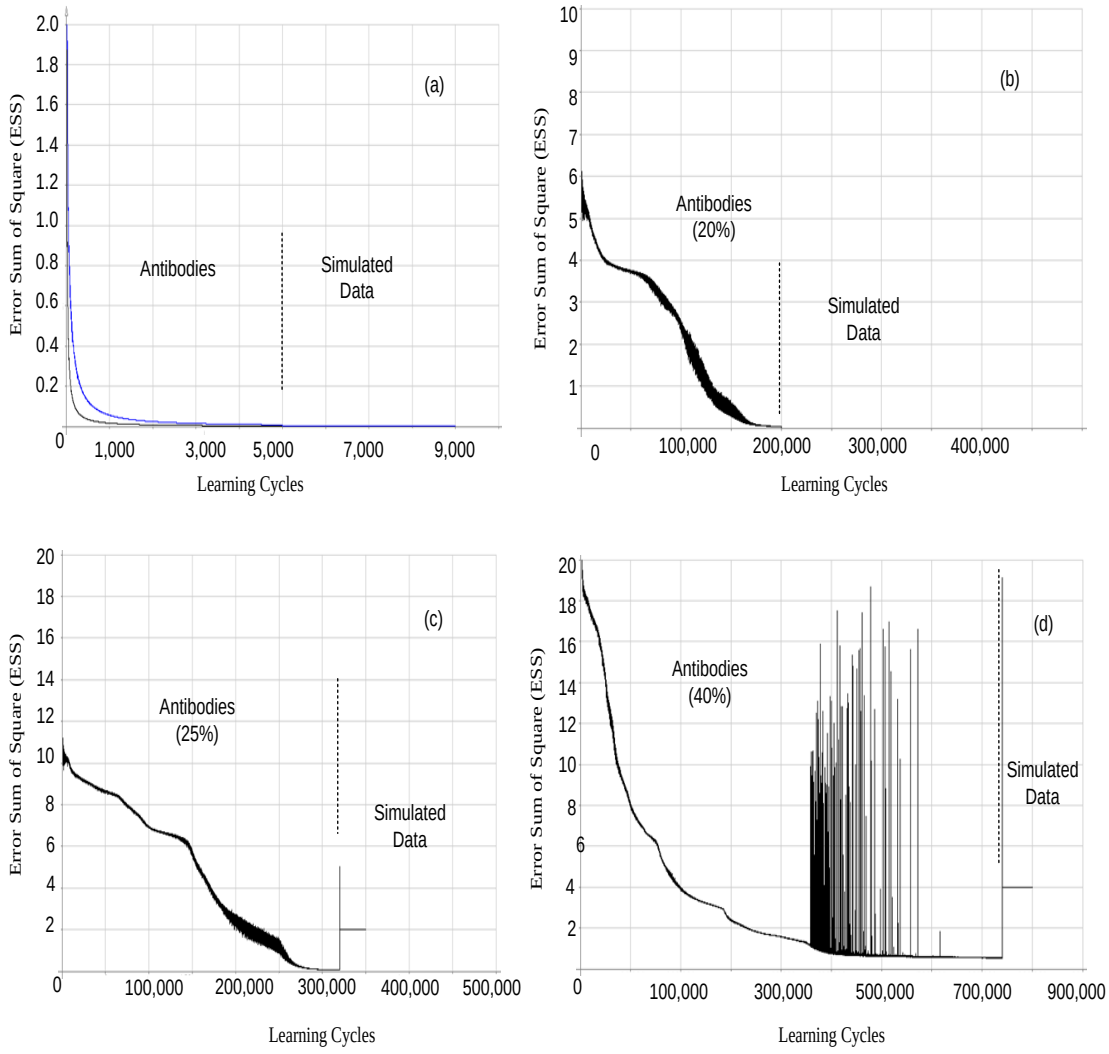


Figure 7.12: Learning error curves of antibodies with different mutation rates. (a): Antibodies with 5% and 10% mutation rates prior to exposure to full simulated data (final convergence at 0.0 ESS). (b): Antibodies with a 20% mutation rate (final convergence at 0.0 ESS). (c): Antibodies with a 25.0% mutation rate (final convergence at 2.0 ESS). (d): Antibodies with a 40% mutation rate (final convergence at 4.0 ESS).

Table 7.3: 10 runs of antibodies (Abs) using different mutation rates on simulated data

Index	1	2	3	4	5	6	7	8	9	10	Avg.	Min.	Max.
Abs 5%	0	0	0	0	0	0	0	0	0	0	0	0	0
Abs 10%	0	0	0	0	0	0	0	0	0	0	0	0	0
Abs 20%	0	0	0	0	0	0	0	0	0	0	0	0	0
Abs 25%	3	1	1	1	3	1	0	1	1	3	1.5	0	3
Abs 40%	4	2	0	2	4	3	2	1	3	4	2.5	0	4

So far we have successfully demonstrated the effectiveness of the vaccination process in achieving better learning capabilities when no prior immune system is present. We still have to investigate the effectiveness of vaccination in cases where some sort of immune system already exists. This is our first step towards providing vaccination in online learning systems. For this purpose, we have divided the datasets into two groups, namely training data and test data. The test data is used to generate vaccination (i.e. generations of both memory cells and antibodies). The steps used to implement C3 are listed in Figure 7.4. In P-I training data was used directly to train the ANN, which represents a form of a primary (or existing) immune system. In P-II memory cells or antibodies along with training data were introduced into the ANN for learning. Once the ANN converged, the test data was introduced to determine the learning capabilities of the ANN. The results obtained using C3 were tested using the same three phase methodology, but without using memory cells or antibodies. When no memory cells were used in the three phase methodology (non-vaccinated three-phase methodology), in P-I the training data was used to train the ANN and in P-II both the training and test data were used to train the ANN. Once there was convergence in P-III, only the test data was used to determine learning capabilities.

7.3.3 Experiments with C3

The original Iris data was randomly divided into two groups: training data and test data (75 instances each). Figure 7.13 (a) shows what happens in a non-vaccinated ANN. Three phases – training data, followed by training + test data, and finally test data – were used. The ANN converged to zero for the training data, but when both the training and the test data were used, the error curve converged to 2. Finally, when only the test data was used, the ANN stayed at 2 errors without any oscillation. Figure 7.13 (b) shows what happens when memory cells are used along with the training data in P-II; the ANN converged to zero error. However, once the test data was introduced, the error curve converged to 2 again.

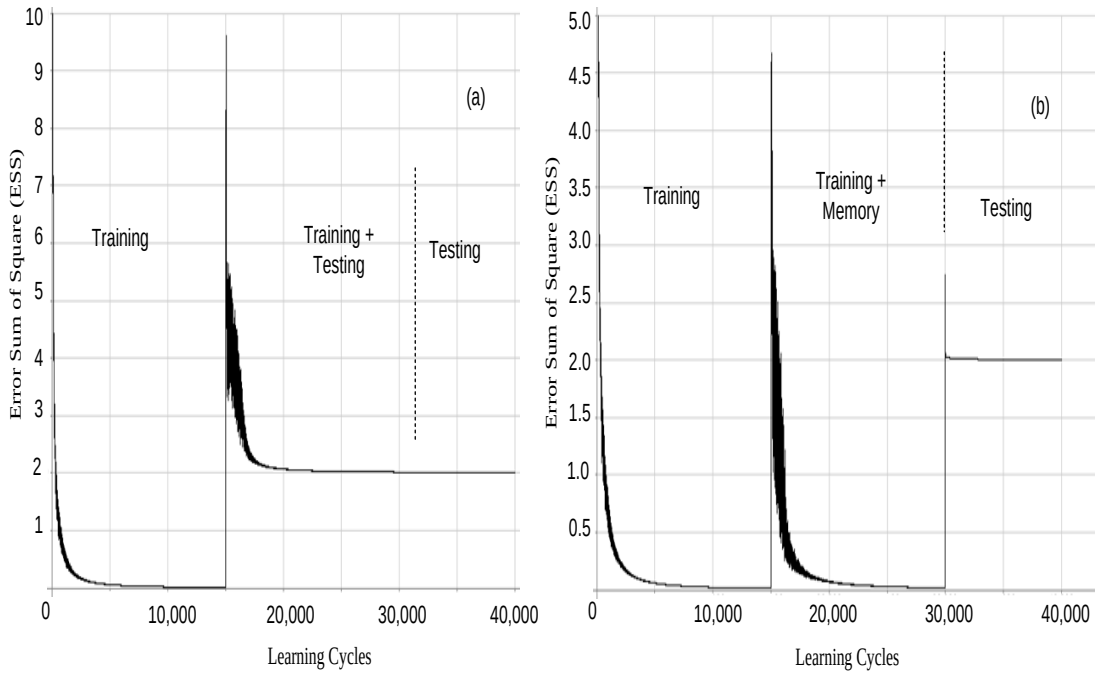


Figure 7.13: Learning error curves of C3. (a): Training and test data are used (final convergence at 2.0 ESS). (b): Memory cells are used instead of training data in P-II to train the ANN (final convergence at 2.0 ESS).

The same experiment was conducted by replacing memory cells with antibodies with different mutation rates. The final results can be seen in Figure 7.14. Antibodies with a 5% mutation rate produced 2 errors on the test data, whereas antibodies with a 10% mutation rate produced zero errors on the test data. The errors increased with antibodies of 15% and 20% mutation rates.

The Breast Cancer Wisconsin data was divided into 369 training and 200 test instances. A test error of 3 was recorded when the non-vaccinated ANN comprising training data (P-I), training + test data (P-II) and test data (P-III) were used (Figure 7.15 [a]). The same test error of 3 was obtained when the ANN was vaccinated with a combination of training data + the memory cells used in P-II (Figure 7.15 [b]).

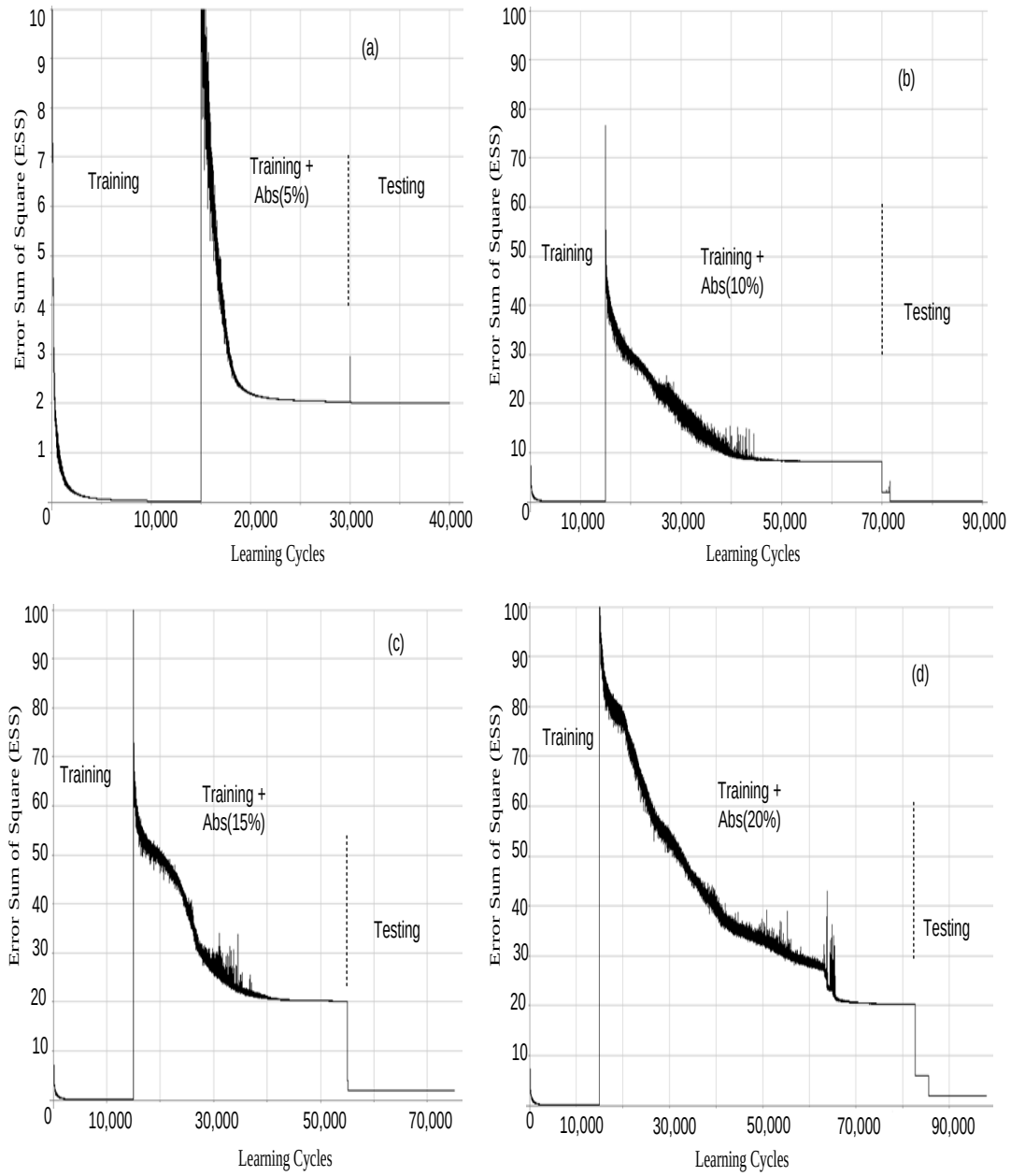


Figure 7.14: Training data (Breast Cancer Wisconsin data) was used to train the ANN in P-I; antibodies with different mutation rates were used along with the training data in P-II; and finally test data was introduced in P-III. (a): The 5% mutation rate converged to 2.0 ESS. (b): The 10% mutation rate converged to 0.0 ESS. (c): The 15% mutation rate converged to 2.0 ESS. (d): The 20% mutation rate converged to 2.0 ESS.

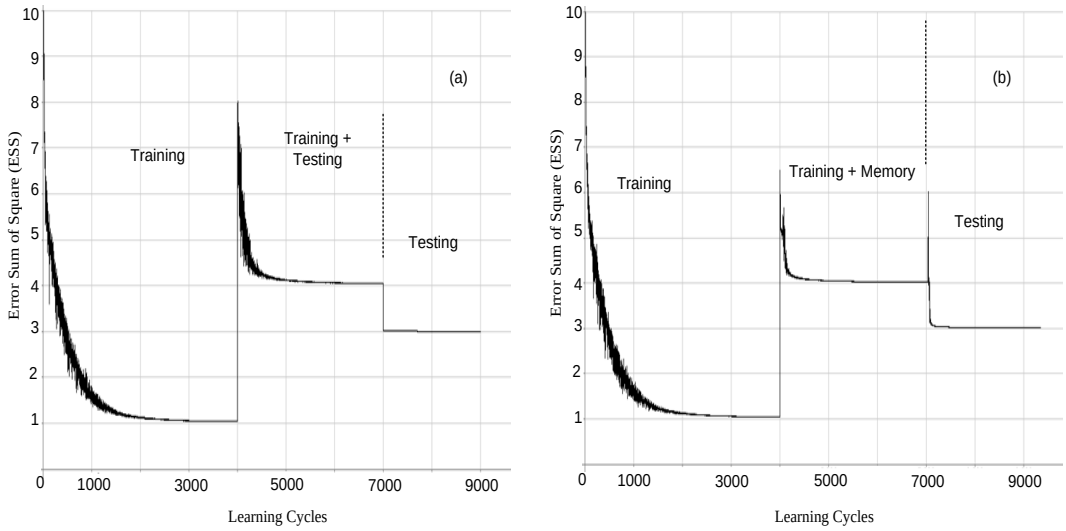


Figure 7.15: (a): Training and test data used in P-II (final convergence of ANN was at 3.0 ESS). (b): Training and memory cells were used in P-II (final convergence of ANN was at 3.0 ESS).

The same experiment was conducted with antibodies produced using different mutation rates in P-II along with the training data (see Table 7.4). Antibodies with a 5% mutation rate produced 3 errors (on average) on test data, whereas 10% mutation resulted in 1.6 errors (on average). Antibodies with 15% and 20% mutations produced 1 and 2 errors on average, respectively, on test data. The results of all 10 runs with different mutation rates can be seen in Table 7.4, where the same population of antibodies was used in each run. The results indicate that the best mutation rate for antibodies is 15% and any other mutation rate above or below results in more learning errors.

Table 7.4: 10 run of antibodies (Abs) using different mutation rates for the Breast Cancer Wisconsin data

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**	**Min.**	**Max.**
Abs 5%	3	3	3	3	3	3	3	3	3	3	3.0	3	3
Abs 10%	1	1	3	3	1	1	3	1	1	1	1.6	1	3
Abs 15%	1	1	1	1	1	1	1	1	1	1	1.0	1	1
Abs 20%	2	2	2	2	2	2	2	2	2	2	2.0	2	2

For the Parkinson's Disease dataset, the same configuration of 45 training and 150 test data instances was used. A learning error of 4 was recorded when a non-vaccinated ANN comprised of training data (P-I), training + test data (P-II) and test data (P-III) was used (see Figure 7.16 [a]). Improved learning was observed when a lower classification error of 1 was obtained when a vaccinated ANN with the combination of training data +

memory cells was used in P-II (Figure 7.16 [b]). No further improvement was observed when antibodies with 5% and 10% mutation rates were used (Figure 7.16 [c] and [d]). Finally, a zero learning error on test data was obtained when antibodies with a 15% mutation rate were used (Figure 7.16 [e]).

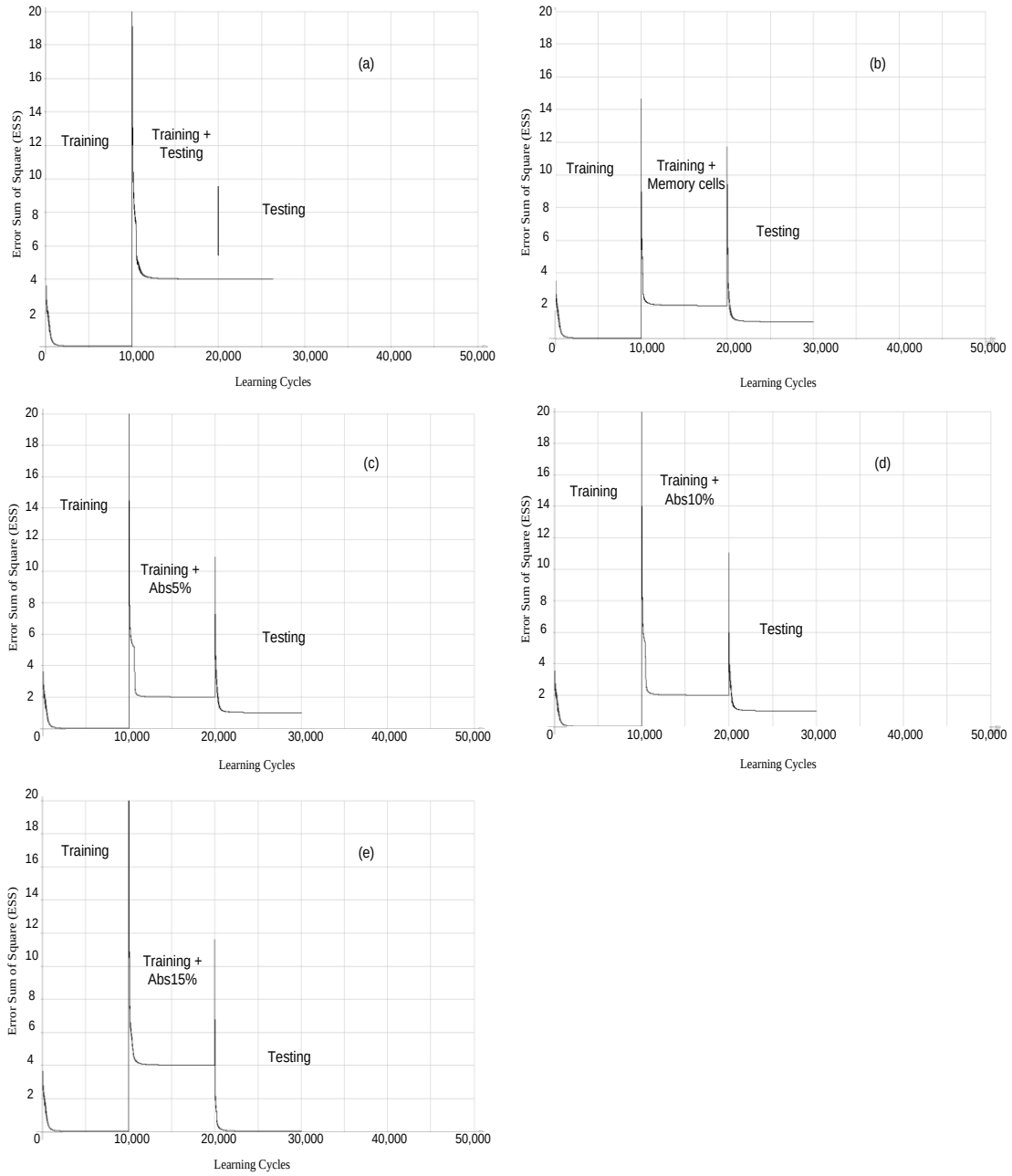


Figure 7.16: Training data was used to train the ANN in P-I. (a): Test data along with training data in P-II; final convergence of ANN when test data was introduced was at 4.0 ESS. (b): Memory cells along with training data in P-II; final convergence of ANN when test data was introduced was at 1.0 ESS. (c, d and e): Antibodies with different mutation rates (5%, 10%, 15%) were used along with training data in P-II; when the test data was then introduced in P-III the final ESSs were 1.0, 1.0, and 0.0 respectively.

7.4 Summary

AIS is said to contribute novelty in at least two ways: data reduction through memory cells and vaccination through antibodies derived from memory cells, with varying rates of hypermutation. The aim of this chapter was to explore these claims for novelty through a systematic series of experiments using an ANN as a 'virtual learning organism' in which a particular AIS algorithm – the HAIS algorithm for supervised learning – is embedded, resulting in a novel hybrid architecture. We have shown by experiments that activation of immune responses can be explained through the mutation rates by which antibodies are generated. If the mutation rate is too low, antibodies cannot adapt to the full range of pathogens in their scope and will suffer from under-generalization in terms of memory cells. On the other hand, if the mutation rate is too high, the immune response is over-activated and antibodies start to overlap in feature space, resulting in over-generalization of data. In this chapter, we have proposed that under-generalization and over-generalization of data can be understood in immunological terms as immunosuppression and autoimmune disease respectively.

Memory cells are an essential part of any AIS algorithm; in our experiments we have shown that memory cells possess excellent data summarization/reduction capabilities for fitting the data. Memory cells prime the ANN in such an efficient manner that when the real data (pathogens) attack the ANN, it is well prepared to handle those pathogens. The main focus of this chapter was to integrate two nature-inspired techniques, an ANN and an AIS in a novel hybrid architecture, where the ANN is the virtual learning organism and the AIS is the embedded immune system engine that both primes and, for the most part, vaccinates the ANN against possible future attack by pathogens. The ANN is originally passive: it only learns what the AIS primes or vaccinates it. The next stage of this work will be to allow the ANN to feed back the results of its learning to the AIS to fine-tune the immune system further, thereby directing the AIS to find even better solutions when confronted by hard-to-categorize samples.

In addition, it was shown through empirical findings that there is a trade-off between high degrees of generalization of data through the random generation of antibodies and the final learning error obtained. A more sophisticated mutation function could be added to help decrease learning error and at the same time improve data generalization. We have shown in this chapter that random mutation through generation of antibodies can

help find more generalized classification results. A more directed mutation function could be devised which takes into consideration the current state of the data (memory cells). An advanced mutation function that can integrate knowledge learnt from existing data patterns as well as some degree of randomness in building antibodies is worth exploration. The AIS algorithm used in this chapter is for supervised learning.

Chapter 8

Conclusion and Future Work

8.1 Overview

In order to establish the basis and underlying scope of this thesis, which proposes a novel artificial immune system (AIS) algorithm, background information regarding the natural immune system (NIS) – its important principles, properties, functionalities and the core theories developed in recent years from a computational perspective – was presented in chapter 2. An extensive literature review based on existing immune system-inspired algorithms was also presented in the same chapter. The current state of AIS and the way forward as proposed by Hart and Timmis [10] were also discussed. Then some of the drawbacks of existing AIS algorithms were highlighted before NISs were outlined at a more detailed biological level (the relationship between immunoglobulins (Igs) and antibodies, the role of memory cells and plasma cells). It was this that led us to the development of a novel AIS algorithm we call the Humoral-Mediated Artificial Immune System (HAIS), inspired by the adaptive immune system and presented in chapter 3. HAIS derived its inspiration from the way adaptive immune system actively produces antibodies once it encounters a pathogen. One advantage of the HAIS algorithm is its capability of finding natural groupings in the data (clusters) as well as outliers. Chapter 3 was divided into two parts: part I demonstrated the effectiveness of the HAIS clustering algorithm using simulated and real-world datasets and part II investigated the capabilities of HAIS towards finding outliers. The parameters of the HAIS algorithm were also evaluated in chapter 3.

The behavior of the HAIS algorithm is stochastic as its clustering results are highly influenced by antigen presentation order. One advantage of going into more detail is that it allows the introduction of other existing evolutionary techniques in a novel way to represent processes that would otherwise be superficially presented. In chapter 4, a population-based HAIS algorithm was proposed to evolve a population of clustering solutions and hence to obtain better clustering solutions. At the micro-level the HAIS algorithm was used to generate clustering solutions while at the macro-level a genetic algorithm (GA) was used to evolve those clustering solutions towards a user-defined fitness function. We demonstrated in chapter 4 that an effective clustering solution can be evolved using HAIS and a standard GA, where incremental transfer of memory cells is used to transfer knowledge from one generation to another (a form of reinforcement learning).

In the HAIS algorithm, one of the most important components is the population of antibodies. Normally, antibodies are cloned and hypermutated copies of already captured antigens. The HAIS algorithm followed a decentralized process, where antigens or the data samples were trapped through generated antibodies, using a local affinity measure between the antigens and the most stimulated antibodies. In chapter 5 we investigated the effectiveness of a hypermutation operator in generating antibodies as well as affinity measures in the context of proposed HAIS algorithm. A three-step methodology was developed to investigate the effects of various mutation rates, where cluster starting points and antigen presentation order were kept fixed and the effectiveness and functionality of the hypermutation operator was evaluated. The experimental results demonstrated that both affinity measure and hypermutation play an important role in finding better clustering solutions.

Chapter 6 extended the idea of the humoral-mediated inspired AIS unsupervised clustering algorithm (HAIS) by applying it to supervised learning. The supervised learning version of HAIS also uses core immune system concepts such as plasma cells, memory cells, antibodies and B-cells, as well as parameters such as the affinity measure and negative clonal selection thresholds. It is a one-shot algorithm where antibodies perform two main functions: (a) capture future antigens (patterns) based on already captured antigens (existing patterns) and (b) stimulated antibodies can learn and update the existing state of learning as new antigens are introduced. The final output of the HAIS supervised learning algorithm was a set of memory cells where the k-nearest

neighbors (NN) method was used to predict class labels on unseen data instances. The work reported in chapter 6 showed that various antigen presentation orders can produce different classification results.

AIS algorithms including HAIS are known to possess data reduction capabilities through memory cells. However, there is currently very little understanding of how effective the memory cells are at reducing the data (i.e. for generalizing to important structural properties of the data) while at the same time preserving critical class information. Again, by going into more detail, we have found that it is possible to introduce novelty in the form of a hybrid architecture to reflect the more complex processes (in this case, the role and function of memory cells) for increased learning. In chapter 7, we demonstrated using a hybrid architecture (HAIS and an artificial neural network [ANN]) that memory cells obtained using HAIS have excellent data reduction capabilities. Furthermore, we successfully established that effective learning can be achieved in an artificial learning system such as an ANN through an artificial vaccination process. Vaccination material is extracted from the original data using the HAIS algorithm in the form of memory cells, which subsequently produces antibodies. These antibodies are used to train the ANN prior to training it on original the data. Learning through vaccination is a novel concept that only came about in our research because of our deeper exploration of the biological role of memory cells.

8.2 Contribution of This Thesis

We began this thesis by proposing the following two research questions:

Q1: *Is it possible to develop an intelligent and biologically plausible learning algorithm inspired by the processes and metaphors of a NIS, informed by the latest scientific research?*

Q2: *Is it possible to incorporate NIS concepts and metaphors with other well-established nature-inspired techniques to achieve efficient and improved learning capabilities?*

These research questions are expressed in general terms. We have developed two immune system-inspired learning algorithms based on a deeper exploration of immune systems. Furthermore, we have coupled immune system metaphors and processes with

well-established ANNs in a novel way to improve learning capabilities. The detailed contribution of this thesis and research therein is:

1. An immune system-inspired unsupervised clustering algorithm called the Humoral-Mediated Artificial Immune System (HAIS) was developed with which natural clusters and outliers can be identified simultaneously. The HAIS algorithm was inspired by NIS components such as B-cells, memory cells, antibodies and it therefore has biological plausibility.

2. A multi-layered AIS model has been developed that provides a more effective memory cell role. In our proposed unsupervised clustering algorithm, memory cells play an active role in capturing antigens (data samples), rather than only being used to classify unseen data.

3. In the HAIS algorithm, learning is mainly performed through antibodies. Antibodies learn from existing patterns (captured antigens) to fine-tune their response in case the same antigens attack again in the future, and future patterns are predicted based on the existing information. In this thesis we have demonstrated with experimental results that conventional affinity maturation (random fine-tuning of antibody receptors to the antigen) is not required. A more deterministic approach such as 100% learning of stimulated antibody receptors to the antigen can be implemented for both supervised and unsupervised HAIS algorithms.

4. In most of the existing AIS models, a static affinity measure is used to build the model. In our proposed HAIS algorithm, we have introduced a dynamic affinity measure which is based on the information present in B-cells.

5. A population-based AIS algorithm is proposed by integrating the standard HAIS and a GA. The incremental transfer of memory cells from one generation to another is helpful in achieving convergence on population of solutions and also leads to better clustering solutions.

6. We have demonstrated that artificial vaccination can help in the learning process. AIS can help (through generation of antibodies) to generalize data which in turn can help to build a more robust and efficient model.

7. It has also been demonstrated that AIS models can be integrated with other nature-inspired approaches to achieve efficient results in building a more generalized model.

The main contribution of the HAIS algorithm and hence this thesis can be explained in the following statement, where antibodies and antigens are considered to be the current and final state of the AIS system respectively:

'Given the current state of the system, the HAIS algorithm tries to predict a future state, in comparison to other AIS algorithms, where, given a (random) current state of the system, the goal is to adjust the AIS system to the final state.'

In the light of the above statement, the main difference between HAIS and other AIS algorithms (e.g. CLONALG, aiNet) is that HAIS tries to evolve or predict future antigens by considering already captured antigens, whereas in other AIS approaches the current state of the system (antibodies) is evolved in the direction of new antigens.

8.3 Limitations of Our Work

HAIS is a decentralized approach as it captures data samples through the interaction of antibodies with data samples. These antibodies are generated using random mutation with the purpose of efficiently capturing future pathogens and covering more search space. The random mutation is a very basic way of searching the feature space for efficient optimal solutions. One limitation of the work presented in this thesis is that no efforts were made to devise a robust and effective directed mutation strategy that can explore search space more efficiently.

In the experiments conducted (with both the unsupervised and supervised HAIS) the maximum number of features used was no more than 50 (approximately). In this thesis no substantial work is presented that could demonstrate the scalability of the HAIS algorithm to very large datasets with thousands of features and millions of instances. Furthermore, the thesis has not covered issues such as feature selection and feature weighting, which are critical problems in the field of machine learning.

Mutation plays an important role in the HAIS algorithm, as exploration and exploitation of the search space are carried out using different mutation rates. A strategy comprising of static mutation rate (pre-defined mutation rate) was used throughout the thesis to

explain the role of the mutation operator in the HAIS algorithm. This thesis does not develop any adaptive and self-evolving mutation strategy that possesses the capabilities of finding a balance between exploration and exploitation and identifying the most effective mutation rate.

In chapter 7, random mutation was used to generate antibodies from memory cells. This random mutation was able to find better classification results due to data generalization. But the problem of data over-generalization arose when a higher mutation rate was used and the classification accuracy of the final generated model deteriorated (see Figure 7.12). The work presented in this thesis does not address this data over-generalization issue.

The question of how to extract information or knowledge from B-cells or memory cells is also not addressed in this thesis.

8.4 Future Work

In a standard HAIS algorithm (chapter 3), directed mutation based on the sum of the variance of each feature is used to find the mutation rate for each feature. A more robust mutation strategy is required that can consider the trade-off between exploration and exploitation (as well as the same exploration/exploitation balance among iterations). A potential mutation strategy could be based on a hybrid of both within-cluster variation (or feature weighting) and random mutation.

In a population-based HAIS approach (chapter 4), we used uniform mutation and a crossover operator at micro- and macro-level processes respectively. We have demonstrated that better clustering results can be achieved using both mutation and crossover operators. An interesting investigation for future work would be to examine the relationship of these two operators to see if by using higher mutation rates only (removing the crossover operator altogether) we can achieve the same clustering results. Further work is also required to evaluate appropriate population size in comparison to data size or the number of clusters found. Various crossover strategies can also be implemented to analyze and optimize the convergence of the HAIS algorithm. In addition, it would be interesting to investigate various memory cell incremental rates to find an appropriate incremental rate required to achieve fast and efficient clustering results.

An affinity threshold (AT) based on the internal structure of each cluster was proposed chapters 3 and 6 for unsupervised and supervised learning algorithms respectively. The method used is justifiable on the grounds that each B-cell in an AIS must evolve according to the internal environment and should not be static throughout. A more sophisticated AT calculating measure needs to be considered in the future, one that can converge automatically to produce better clustering solutions.

In chapter 7 we proposed a hybrid architecture that incorporates two nature-inspired techniques, ANN and HAIS, where the ANN is the virtual learning organism and the AIS is the embedded immune system engine that both primes and vaccinates the ANN against possible future attack by pathogens. The ANN in its current (original) state is passive; it only learns what the AIS primes it or vaccinates it. Future work should investigate enabling the ANN to feed back the results of its learning to the AIS to fine-tune the immune system further, thereby directing the AIS to find even better solutions when confronted by hard-to-categorize samples.

In chapter 7 the role of antibodies in terms of various levels of immune response to captured pathogens (using various mutation rates) was extensively discussed. It was concluded that if the mutation rate is too low, antibodies cannot adapt to the full range of pathogens in their scope and will suffer from under-generalization (or over-specialization) in terms of memory cells. On the other hand, if the mutation rate is too high, the immune response is over-activated, resulting in over-generalization of data. Future work is required to select a more robust mechanism that can generate a self-evolving and self-organizing immune response by generating a population of antibodies that can remove the need of manually selecting a mutation rate. A more directed mutation function could be devised which takes into consideration the already seen data instances (memory cells) and randomness (or bias) in the current state of a learning system. In short, we are suggesting that a directed mutation operator that can integrate knowledge learnt from existing pathogens (data patterns) as well as some degree of randomness in order to generate a population of antibodies is worth exploration.

In the future, rigorous analysis will be required on the different parameters proposed in the HAIS unsupervised and supervised algorithms. For example, the effects on the populations of B-cells, antibodies (produced by B-cells) and memory cells by the different HAIS parameters need further investigation. The variations in experimental

results and comparisons of the effects of different parameter values can be analyzed using t-tests, analysis of variance (ANOVA) and non-parametric quartile testing (box and whisker plots) [194]. For the comparison of the HAIS supervised algorithm with the Artificial Immune Recognition System (AIRS) and other AIS-inspired algorithms, extensive experiments are required, and these should include not only accuracy comparisons, but other statistical measures such as sensitivity and specificity analysis as well [195]. Furthermore, the Kappa statistic (or Kappa coefficient) [196] can be used to evaluate the performance of different AIS-inspired classifiers with the HAIS supervised algorithm. To determine the empirical computational complexity of the HAIS algorithms, the CPU time taken to generate a model can be investigated and then compared with existing AIS and non-AIS classifiers. These experiments must be done under the same conditions using the same datasets to preserve comparability.

Supervised and unsupervised learning is a very narrow domain for the applications of these kinds of nature-inspired, evolvable algorithms. Such algorithms can also be used for other hard problems such as optimization and subset selection. A broader scope of such research work in the future may include wetware, or synthetic biology (artificially created cellular processes). That is, sometime in the future, wetware technologists may be able to create their own synthetic immune systems using different principles from existing NISs in order to keep the wetware systems self-organized and self-sustaining.

Finally, we have left unexplored the question of what going into more detail with regard to immune systems actually means. We are still not at the level of biochemical reactions, nor do we as computer scientists wish to go down to that level. Instead, our task has been to devise novel algorithms based on a ‘deeper understanding’ of the biology of immune systems. But it is not clear what the principles and limits are of a ‘deeper understanding’, given the absence of a standardized and formal framework for describing AISs. The final part of this chapter will attempt to explore the notion of ‘deeper understanding’ in terms that a computer scientist may understand, that is, in terms of a theoretical framework. We will also attempt to demonstrate the benefit of this ‘deeper understanding’ through a final set of experiments in a domain of great importance to theoretical researchers: optimization.

8.5 Theoretical Advances

8.5.1 Variable AT for Each B-cell

In previous AIS approaches to clustering, the AT parameter is calculated beforehand and remains fixed throughout the clustering process. In addition, all clusters (B-cells) are assigned with the same AT. In chapter 3 of this thesis, we proposed a dynamic and self-evolving AT for each of the B-cells. As far as we are aware, this is the first demonstration of an evolving AT measure and shows that B-cells can be interpreted as independent 'agents' that can learn from their environment. Another parameter used in the HAIS algorithm is network threshold (NT) which is responsible for the mergence of similar B-cells (clusters). Both AT and NT started with the same value but AT increases, whereas NT decreases, with increasing numbers of iterations. Below is the equation used to set the initial AT and NT values based on the initial selection of antigens.

$$AT = NT = \frac{\frac{1}{c}\sum_{i=1}^{c}\sigma_i}{\alpha} \qquad (1)$$

where c is the number of features in the data and σ is the standard deviation of each feature. The parameter α is a scalar value which controls the tightness of boundaries among the clusters. AT and NT are updated as follows:

$$NT = NT.nt_{rate} \quad where \;\; nt_{rate} \leq 1 \qquad (2)$$

$$AT = \beta.new_{aff} + \gamma.Bcell_{aff} \qquad (3)$$

$$where \quad \beta + \gamma = 1$$

$$new_{aff} = AT.at_{rate} \quad where\ at_{rate} \geq 1 \qquad (4)$$

$$Bcell_{aff_j} = \left(\frac{1}{f}\sum_{i=1}^{f}\frac{\sigma_i(Bcell_j).\mathrm{L}}{\alpha}\right).at_{rate} \qquad (5)$$

where L is the number of clusters.

In other words, the above equations express a mechanism whereby learning in a B-cell through AT is initially performed uniformly but later, upon each complete exposure of antigens, the AT is updated according to the interaction of the B-cell with captured antigens and NT is decreased based on nt_{rate}. One of the advantages of this mechanism

is that outlier detection becomes possible. That is, instead of forcing antigens to be allocated to a predefined number of B-cells (clusters), B-cells only capture antigens that are within their neighborhood. Therefore, HAIS can recognize meaningful clusters as well as outliers in the data simultaneously.

New B-cells that are released into the body are called naive B-cells. The life span of these naive B-cells is very short. They circulate in the body and any cell that cannot stimulate or capture antigens is removed from the immune system. Our HAIS approach to unsupervised learning also uses this principle to evaluate the stimulation or activation level of all existing B-cells. In the HAIS algorithm we introduced a death threshold (DT) parameter to simulate this process. At the end of each cycle, B-cells are evaluated based on the number of antigens each has captured. Any B-cell that has captured fewer antigens than a certain threshold (DT) is removed. This is helpful in promoting natural competition among B-cells for capturing antigens more effectively, so that they can survive (through natural selection) into the next generation. A B-cell that does not attract enough antigens is disposed of, but there is no restriction on the generation of new B-cells during the next generation. As far as we are aware, this is the first time that the relationship between DT and effective clustering has been demonstrated.

8.5.2 More Effective Role of Memory Cells

In a NIS, the presence of memory cells plays a significant role in fighting against infections and diseases. NISs can mount a more effective and faster response to already seen antigens due to the presence of memory cells. In previous AIS approaches, memory cells, if used at all, are used for finding a class belonging of unseen data (test data) and for data compression/abstraction purposes. In this thesis, we have proposed a more effective role for memory cells in both HAIS supervised and unsupervised learning models. In our approach, memory cells play an active role, not only at the test phase but also during the training phase, where memory cells are used to capture already seen antigens (data instances).

In AIS approaches, memory cells represent information learned during the current learning cycle and transferring that knowledge is essential for incremental learning. In our population-based HAIS approach, the concept of incremental learning is re-expressed through incremental transfer of memory cells from one generation to another (chapter 4).

In most AIS approaches, a fixed repertoire (pool) of antibodies is kept, and stimulated antibodies are evolved in the direction of newly invading antigens. If learning is one-dimensional, the AIS 'forgets' previously learnt antigens if the antibody repertoire is kept fixed [64]. But in machine learning this aspect can have serious repercussions, especially since new clusters develop and grow in size constantly. In our HAIS algorithm, knowledge learned previously is not 'forgotten': the population of antibodies and memory cells increases as new pathogens are encountered. On the other hand, this ever-increasing size of the population of antibodies also has computational disadvantages. Here, the important question for future AIS research is: Is it better to completely 'forget' something learned in the past (past experience) or do we somehow keep some information already gained? The former may be better for computational efficiency and the latter for computational effectiveness.

8.5.3 Homeostatic State

All natural systems tend to stay in a stable state called homeostasis [33]. This homeostatic state is achieved through the interaction of various organs and cells constituting the natural system (intra-system homeostasis). On the other hand, inter-system homeostasis is achieved when various systems interact with each other to perform various complex tasks to keep the organism alive and healthy. In this thesis, we have shown that intra-system and inter-system homeostasis can be maintained using the HAIS algorithm and the population-based HAIS algorithm respectively. In chapter 3, the HAIS algorithm starts with a random numbers of clusters and after a certain number of iterations converges to a stable and fixed number of clusters; this can be seen as a state of intra-system homeostasis. On the other hand, in the population-based HAIS approach (chapter 4), the whole population converges to stable and local optimal clustering solutions after a number of generations through the transfer of knowledge from one generation to another. This process can be seen as inter-system homeostasis, where the evolutionary approach in conjunction with AIS concepts is used. More work is required in the future to relate AIS stability and steady states with homeostasis theory and concepts, since it is likely that formal models of AIS computability will need to refer to dynamic and stable AIS states.

8.5.4 Effects of Various Antigen Presentation Orders

In previous AIS approaches, a learning algorithm generally consists of two phases. Phase 1 consists of generating candidate memory cells based on recognizing randomly presented antigens. In Phase 2 those candidate memory cells compete for the final pool of memory cells. In other words, in Phase 2 the spatial positioning of the generated memory cells is optimized based on already seen antigens. Therefore, memory cells are not allowed in the capturing of pathogen at the training phase. In our approach, we have assigned a more active role to memory cells, which capture antigens at the training phase. We have removed the memory cell optimization phase at the micro level. This removal of the memory cell optimization phase has presented a different challenge in a way that now random order of pathogen presentation have started to construct the final learning model differently. To address this issue we have proposed a population-based AIS approach that optimizes the spatial positioning of memory cells by allowing various orders of antigen presentation. With this technique we have successfully constructed a state of homeostasis at the population level, where the population of solution given number of generations started to converge to local optimal solutions.

In addition, in previous unsupervised learning approaches, the focus has been to extract a summary of the original data in the form of memory cells, and later those memory cells are allocated to different clusters or classes based on some similarity or dissimilarity measure [74, 79]. In our HAIS unsupervised learning algorithm, we have assigned cluster association (a label) to antigens at the point of capture. In an NIS, when an antigen enters the body, the NIS performs two actions: (1) recognition of pathogen and (2) mounting of an appropriate immune response. Therefore, our HAIS algorithm is more ‘faithful’ to an NIS in that it assigns a label to any antigen once it is recognized as belonging to a certain cluster.

8.5.5 Behavior of AT and Mutation

AT and mutation are important parameters of any AIS algorithm. There has not been much research in the behavior of these parameters or on how much these parameters affect the learning of AIS algorithms. In chapter 5 we demonstrated that given a fixed starting point of each cluster (B-cell), AT and mutation follow the same behavior (in terms of clustering accuracy). The higher the mutation rate or AT parameter, the greater the oscillation in the clustering results (accuracy), and vice versa. Here, greater

oscillation in clustering results means high fluctuations in the clustering accuracy when the obtained class labels are compared against the true class labels.

Given a fixed number of B-cells (clusters), for a B-cell to capture fewer similar antigens there exist two options: (1) increase the AT measure; or (2) increase the mutation rate. Option 1 would decrease the similarity criteria so that even fewer similar antigens are accepted. Option 2 generates antibodies with a higher mutation rate that can cover more feature space and can capture antigens located further from the cluster center. To implement Option 2, various mutation rates are used (for more details see chapter 5). On the other hand, to implement Option 1, we start with an initial value of the affinity threshold that is defined by the following expression:

$$AT = \left(\frac{1}{N}\sum_{i=1}^{f} s(X_i)\right) * \alpha \qquad (6)$$

where s is the standard deviation and X_i is the i^{th} data feature, f is the number of features and N is the total number of instances in the data.

Once all antigens associated with the AT are captured and there are no more antigens left in the antigen pool, then the AT parameter is increased. To allocate fewer similar antigens, AT is incremented as follows:

$$AT_{inc} = \min\ (affinity) + \left(\frac{1}{N}\sum_{i=1}^{f} s(X_i)\right) * \beta \qquad (7)$$

where *affinity* is the distance measure between B-cell antibodies and antigens and β is another user-defined parameter.

Our experimental results showed that both the AT and the mutation rate parameters play an important role in finding improved clustering results (higher clustering accuracy). Using only the AT parameter, one can find reasonably good clustering results. However, the mutation rate plays an important role in further improving the clustering results obtained given a certain AT parameter (for more details see chapter 5). Future work is needed to explore the relationship between antigen-to-antigen similarity and its relationship to cluster-to-antigen similarity in more detail so that appropriate parameter

values can be set for different datasets. We explored this aspect to some extent in this thesis, and this is discussed in the next section.

8.5.6 Variable AT Measure HAIS Supervised Algorithm

Watkins [197] stated that a variable mutation rate can be used in supervised learning models to obtain a generalized AIS supervised model by keeping the AT constant. In chapter 5 we showed with experimental results that both AT and mutation rate behave in a similar fashion. Therefore, in chapter 6 we proposed a HAIS algorithm for supervised learning that uses a variable AT parameter value while keeping the mutation rate constant. The experimental results in chapter 7 suggest that the final results obtained – in terms of both classification accuracy as well as capabilities of data summarization – are as good as those for the AIRS algorithm.

Both the equal and variable AT parameter values calculated in HAIS are described below. Equation (8) below calculates equal AT for each class (in the data) by considering the sum of the standard deviation of each feature in the data and then dividing it by a number of features in the data. The resulting value is then divided by a user-defined parameter α to make the affinity measure stronger or weaker among antigens and antibodies while calculating similarities. On the other hand, in Equation (9) each class is assigned a different AT depending on its own (class) standard deviation. In other words, the sum of the standard deviation of the data related to each class is calculated separately, and then divided by the number of features to calculate the local AT value for each class.

$$AT = \frac{\frac{1}{f}\sum \sigma\,(data)}{\alpha} \qquad (8)$$

where σ is the standard deviation of the data and α is the user defined parameter which must be a non-negative and greater-than-zero value.

$$AT = \frac{\frac{1}{f}\sum_{i=1}^{labels} \sigma(data^i)}{\alpha} \qquad (9)$$

where σ is the standard deviation of the data, and $data^i$ is all data instances associated σ with class label i.

8.5.7 Affinity Maturation

In previous AIS approaches, a random mutation of an antibody receptor is used to achieve affinity maturation. In this thesis, we demonstrated with experimental results that the higher the affinity maturation, the better the results (chapter 6). Therefore, we proposed the removal of the random fine-tuning of antibody receptors towards antigenic receptors. Instead, we performed 100% transformation (fine-tuning) of antibody receptors towards the captured antigen. In other words, we have demonstrated that a form of directed evolution can work effectively and efficiently in an AIS.

8.5.8 HAIS for Optimization

The objective of this thesis, as explained in chapter 1, is to develop novel supervised and unsupervised learning algorithms using inspiration from the humoral-mediated immune response triggered by an adaptive immune system. Put another way, this thesis investigates AISs for optimization purposes. That is, HAIS is essentially a set of optimization mechanisms and techniques that refine B-cells to capture data in an increasingly optimal way. It should therefore not be surprising if HAIS can also be used to deal with classical optimization problems. Although not part of the thesis itself, we include some work that demonstrates that this interpretation of HAIS has some merit in appendix B. An HAIS-optimization algorithm, generally speaking, has the following characteristics:

- Performs effective local search by the evolving population of B-cells through generation of antibodies (affinity maturation)
- Help to find global optimal solutions by generating the random population of B-cells and adding it to the existing population of B-cells
- Keeping track of already explored paths through the help of memory cells that can avoid repeatedly searching the same search space

Readers are referred to appendix B for details of experiments conducted under this interpretation of HAIS (as an optimization algorithm). This work is not finished, but we have shown that the HAIS has applicability in optimization. Therefore, it can be considered as a topic for future work.

8.5.9 Final Thoughts

To summarize, this thesis has explored several aspects of AIS, rather than just one. In particular, the thesis has taken a humoral-mediated approach to AISs, developing this to the extent that new algorithms (which may be seen as refinements of existing AIS algorithms) have been shown to be effective in the learning domain. Our humoral-inspired clustering algorithm explicitly incorporates the process of antibodies mutating, not by themselves, but within plasma cells that then release the antibodies to capture antigens. In other words, our algorithm is more faithful to the principle of new, not previously encountered antigens being matched against Ig receptors on the surface of B-cells. When antigens stick to the Igs, the B-cell evolves into a plasma cell which releases antibodies that are both faithful copies of the original Igs as well as mutated versions of those Igs. Also, the original, triggered B-cell makes a memory cell so that if the same antigen reappears it can be more quickly captured in future. Our approach also uses the concept of natural killer cells (not explicitly included in our algorithms) to remove any B-cells that are not activated by antigens. Our algorithm therefore introduces new mechanisms not previously used in AIS algorithms as well as novel representations of Igs and antibodies.

In developing novel AIS algorithms that reflect humoral mediation to a greater or lesser extent, we have extended our knowledge of affinity thresholds, the power of memory cells, convergence and stability, the effects of different antigen presentation orders, mutation rates of antibodies, and affinity maturation. Also, we have provided some pointers as to where we believe future research in AIS is heading: computability, formal theory, and optimization.

This thesis probably raises more questions than it answers, since we did not know at the start how open-ended the research would become once humoral mediation was adopted as a driving and inspiring concept. While our understanding of various AIS concepts has been extended, so has the number of unanswered questions. This is a healthy sign that this area is worthy of further exploration and debate. AIS is still a small area of computing research and there may be a tendency for researchers outside this area to consider AIS as a ‘solution looking for a problem’, given that other machine learning techniques already cover many of the areas covered in this thesis.

There are computational advantages in being more faithful to the biology. For a computational antibody to mutate directly means that every antibody must now contain the evolutionary mechanisms required for mutation as well as building the modified receptors. This is redundancy on a grand scale, especially if the antibodies are not successful at capturing antigens and must therefore be destroyed. It is conceptually more elegant and efficient to stick with what we know about how the NIS produces antibodies, which is through B-cells and their plasma equivalents. These cells then contain the necessary mutation and biomolecular machinery to produce antibodies of a possibly infinite variety.

It is 8 years since Garrett asked whether AIS actually offered anything useful over existing nature-inspired approaches, including GAs and neural networks [198]. Garrett provided a number of ways in which 'useful' could be defined in terms of being 'distinctive' and 'effective'. An AIS is distinctive if unique symbols, novel expressions and unique processes result from the inspiration. An AIS is effective if its methods provide a unique means of obtaining a set of results, or produces results that are better than existing results, or produces results more quickly than other methods (at least one of these must be true, according to Garrett, if an AIS is to be effective). Garrett's answer to these questions in 2005 was that AISs were likely to become more useful over the next few years but that there were few applications for which 'it is indisputably the most effective method'.

Our humoral-inspired algorithm satisfies both of Garrett's requirements. It is distinctive in that new symbols are used for representing Igs and antibodies as well as plasma cells. Our algorithm contains novel expressions for matching antigens and Igs on the one hand, and antigens and antibodies on the other. Novel processes include an activated B-cell producing both plasma and memory cells. Our algorithm is stochastic and any stochastic algorithm will be hard-pressed to satisfy Garrett's firm requirement that an AIS produce results better than existing results obtained by deterministic algorithms. However, our experiments so far indicate that the results compare favorably with classical, non-stochastic clustering techniques. An overview of our AIS clustering architecture was provided in Figure 3.1.

One useful side-effect of our approach is that, by sticking more faithfully to the biology, we can now deal with outliers. Visual representations of experimental results in chapter

3 (see Figure 3.6) reveal the presence of outliers and show that outlier detection is a form of oscillatory behavior in our algorithm. The presence of these outlying clusters did not affect the way that the rest of the objects were clustered, which provides an important advantage of our method over other methods. There is no reason to remove these outliers to get better clustering behavior among the remaining objects.

Finally, the importance of our work in the long term may not lie in applications to clustering but in possible use in synthetic cell construction. That is, with increasing interest in both industry and academia concerning functional aspects of systems biology and rational synthesis/engineering of novel cellular structures for gene and cell-based therapies, at some point suitable defense mechanisms will need to be built into synthetic cells used in immunotherapy to protect them from attack, either by the body's own immune system or outside invaders. There is currently very little understanding of how to build suitable defense mechanisms into synthetic cells to ensure their effectiveness in the face of hostile self-defense and external attacks. The possibility of viruses or other pathogens taking over the cellular machinery of synthetic therapeutic cells is strong, given that such cells will typically contain the minimum amount of genetic material to serve their purpose. AIS approaches currently offer the best way to explore novel mechanisms for incorporation in, for example, monoclonal antibodies (cells designed to identify and help destroy antigens and other foreign substances) so that they receive some protection without the need to close down or reduce the efficiency of the NIS.

References

[1] P. N. Tan, M. Steinbach, and V. Kumar, "cluster analysis: basic concepts and algorithms," *Introduction to Data Mining, Addison-Wesley*, pp. 487-568, 2006.

[2] D. E. Goldberg, "Genetic Algorithms in Search Opimization and Machine Learning," *Addison-Wesley, Reading, MA*, 1989.

[3] G. Jones, P. Willett, and R. C. Glen, "Molecular Recognition of Receptor Sites using a Genetic algorithm with a Description of Desolvation," *Journal of Molecular Biology*, vol. 245, pp. 43-53, 1995.

[4] D. S. Goodsell and A. J. Olson, "Automated Docking of Substrates to Proteins by Simulated Annealing," *Proteins: Struct. Funct. Genet.*, vol. 8, pp. 195-202, 1990.

[5] S. Kirkpatrick, C. D. Gelatt, and M. P. V. Jr., "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671-680, 1983.

[6] P. S. Shelokar, V. K. Jayaraman, and B. D. Kulkarni, "An Ant Colony Approach for Clustering," *Analytica Chimica Acta*, vol. 509, pp. 187-195, 2004.

[7] M. Dorigo, M. Birattari, and T. Stutzle, "Ant Colony Optimization: Artificial Ants as a Computational Intelligence Technique," *IEEE, Computational Intelligence Magazine*, vol. 1, pp. 28-39, 2006.

[8] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," *Proc. IEEE Int'l. Conf. on Neural Networks*, vol. IV, pp. 1942-1948, 1995.

[9] D. Dasgupta and F. Gonzalez, "Artificial immune system (AIS) research in the last five years," *Proceedings of the Congress on Evolutionary Computation*, pp. 123 - 130, 2003.

[10] E. Hart and J. Timmis, "Application area of AIS: The Past, The Present and the Future," *Applied Soft Computing, Elsevier Science, Amsterdam*, vol. 8, 2008.

[11] J. E. Hunt and D. E. Cook, "Learning using an artificial immune system," *Journal of Network and Computer Applications*, vol. 19, pp. 189-212, 1996.

[12] L. N. d. Castro, "Fundamentals of natural computing: an overview," *Physics of Life Reviews*, vol. 4, pp. 1-36, 2007.

[13] S. H. Liao and C. H. Wen, "Artificial neural networks classification and clustering of methodologies and applications – literature analysis from 1995 to 2005," *Expert Systems with Applications*, vol. 32, pp. 1-11, 2007.

[14] T. Heskes, "Self-organizing maps, vector quantization, and mixture modelling," *IEEE Transactions on Neural Networks*, vol. 12, pp. 1299-1305, 2001.

[15] A. Narayanan, E. C. Keedwell, S. S. Tatineni, and J. Gamalielsson, "Single-layer neural networks for gene expression analysis," *Neurocomputing*, vol. 61, pp. 217-240, 2004.

[16] G. A. Carpenter and S. Grossberg, "The ART of adaptive pattern recognition by a self-organizing neural network," *Compter*, vol. 21, pp. 77-88, 1988.

[17] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993-1001, 1990.

[18] B. D. Ripley, "Pattern Recognition and Neural Networks," *Cambridge University Press, Cambridge UK*, 1996.

[19] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 23-38, 1998.

[20] K. Hsu, H. V. Gupta, and S. Sorooshian, "Artificial Neural Network Modeling of the Rainfall-Runoff Process," *Water Resources Research*, vol. 31, pp. 2517-2530, 1995.

[21] R. H. Sheikh, A. N. Jaiswal, and N. M. Raghuwanshi, "Genetic Algorithm Based Clustering: A Survey," *First International Conference on Emerging Trends in Engineering and Technology*, 2008.

[22] K. Premalatha and A. M. Natarajan, "A New Approach for Data Clustering Based on PSO with Local Search," *Computer and Information Science*, vol. 1, pp. 139-145, 2008.

[23] Y. Kao and K. Cheng, "An ACO-based clustering algorithm," *Springer Berlin / Heidelberg*, vol. 4150/2006, pp. 340-347, 2006.

[24] Y. Marinakis, M. Marinaki, and N. Matsatsinis, "A Stochastic nature inspired metaheuristic for clustering analysis," *Int. J. Business Intelligence and Data Mining*, vol. 3, pp. 30-44, 2008.
[25] M. Dorigo and G. D. Caro, "Ant Colony Optimization: A New Meta-Heuristic," *Proceedings of the Congress on Evolutionary Computation (CEC)*, vol. 2, pp. 1470-1477, 1999.
[26] Q. Liu, L. Wang, A. G. Frutos, A. E. Condon, R. M. Corn, and L. M. Smith, "DNA computing on surfaces," *Nature (London)*, vol. 403, pp. 175-179, 2000.
[27] G. Paun, G. Rozenberg, and A. Salomaa, "DNA Computing: New Computing Paradigm," *ISBN 3-540-64196-3, Springer-Verlag Berlin Heidelberg, New York*, 1998.
[28] E. Aimeur, G. Brassard, and S. Gambs, "Quantum Clustering Algorithms," *Proceedings of the International Conference on Machine Learning*, vol. 24, 2007.
[29] D. Horn and A. Gottlieb, "Algorithm for Data Clustering in Pattern Recognition Problems Based on Quantum Mechanics," *Physical Review Letters*, vol. 88, 2002.
[30] L. N. d. Castro, "Artificial immune systems as a novel soft computing paradigm," *Soft Computing (Berlin, Germany)*, vol. 7, pp. 526-544, 2003.
[31] D. Dasgupta, "Artificial Immune Systems and Their Applications," *Springer-Verlag, Inc. Berlin*, 1999.
[32] S. Forrest and S. Hofmeyer, "Immunology as information processing," *In Segel, L. and Cohen, I., editors, Design Principles for Immune System and Other Distributed Autonomous Systems. Oxford University Press*, p. 361, 2000.
[33] J. Timmis, "Artificial immune system - today and tomorrow," *Natural Computing*, vol. 6, pp. 1-18, 2007.
[34] T. Mitchell, "Machine Learning.," *MacGraw Hill. ISBN 0-07-042807-7*, 1997.
[35] N. J. Nilsson, "Introduction to Machine Learning," *An Early Draft of a Proposed Textbook*, http://ai.stanford.edu/~nilsson/mlbook.html, 1998.
[36] S. K. Murthy, "Automatic construction of decision trees from data: A multi-disciplinary survey," *Data Mining and Knowledge Discovery*, vol. 2, pp. 345-389, 1998.
[37] J. Quinlan, "C4.5: programs for machine learning," *Morgan Kaufmann, San Francisco*, 1993.
[38] F. V. Jensen, "An introduction to Bayesian networks. ," *London: UCL Press.*, 1996.
[39] I. J. Good, "Probability and Weighting of Evidence," *Griffin, London.*, 1950.
[40] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.
[41] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial Neural Networks: A Tutorial," *IEEE Computer*, pp. 31-44, 1996.
[42] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1967, pp. 281-297, 1967.
[43] G. Celeux and G. Govaert, "A classification EM algorithm for clustering and two stochastic versions," *Computational statistics & data analysis*, vol. 14, pp. 315-332, 1992.
[44] L. Zelnik-Manor and P. Perona, "Self-Tuning Spectral Clustering," *Advances in Neural Information Processing Systems*, vol. 17, 2004.
[45] U. Fayyad, G. G. Grinstein, and A. Wierse, "Information visualization in data mining and knowledge discovery," vol. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
[46] I. K. Fodor, "A survey of dimension reduction techniques," *UCRL-ID-148494, U.S. Department of Energy, Lawrence Livermore National Laboratory*, 2002.
[47] M. Ludl and G. Widmer, "Relative Unsupervised Discretization for Association Rule Mining," *Principles of Data Mining and Knowledge Discovery, LNCS, 2000*, vol. 1910/2000, pp. 148-158, 2000.
[48] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-285, 1996.

[49] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 25-34, 1987.
[50] M. Martin, B. Chopard, and P. Albuquerque, "Formation of an ant cemetery: swarm intelligence or statistical accident? ," *Future Generation Computer Systems*, vol. 18, pp. 951-959, 2002.
[51] P. Grasso, "Essentials of Pathology For Toxicologists," *CRC Press*, 2002.
[52] N. Jerne, "Towards a network theory of the immune system," *Annales d'immunologie*, 1974.
[53] D. H. Wolpert and W. G. MacReady, "No Free Lunch Theorems for Search," *Technical Report SFI-TR-95-02-010. Sante Fe, NM, USA: Santa Fe Institute.*, 1996.
[54] A. K. Jain, "Data Clustering: 50 Years Beyond K-means," *Lecture notes in computer science, Springer, Berlin, Heidelberg*, p. 3, 2008.
[55] A. A. Freitas, "Data mining and knowledge discovery with evolutionary algorithms," *ISBN 3-540-43331-7, Springer-Verlag Berlin Heidelberg, New York*, 2002.
[56] J. Kleinberg, "An Impossibility Theorem for Clustering," *Advances in Neural Information Processing Systems, MIT Press*, vol. 15, pp. 463-470, 2003.
[57] U. Aickelin, "Artificial Immune Systems (AIS) – A New Paradigm for Heuristic Decision Making," *OR46 (2004) AIS for OR, Keynote Speech*, 2004.
[58] W. Ahmad and A. Narayanan, "Humoral-mediated Clustering," *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2010)*, pp. 1471-1481, 2010.
[59] W. Ahmad and A. Narayanan, "Outlier Detection using Humoral-mediated Clustering (HAIS)," *Proceedings of NaBIC2010 (IEEE World Congress on Nature and Biologically Inspired Computing)*, pp. 45-52, 2010.
[60] W. Ahmad and A. Narayanan, " Humoral Artificial Immune System (HAIS) For Supervised Learning," *Proceedings of NaBIC2010 (IEEE World Congress on Nature and Biologically Inspired Computing)*, pp. 37-44, 2010.
[61] W. Ahmad and A. Narayanan, "Principles and methods of Artificial Immune System Vaccination of Learning Systems," *Pietro Liò, Giuseppe Nicosia, Thomas Stibor (Eds.): Artificial Immune Systems, 10th International Conference, ICARIS 2011, Cambridge, UK, Lecture Notes in Computer Science 6825 Springer 2011*, pp. 268-281, 2011.
[62] W. Ahmad and A. Narayanan, "Population-Based Artificial Immune System Clustering Algorithm," *Pietro Liò, Giuseppe Nicosia, Thomas Stibor (Eds.): Artificial Immune Systems, 10th International Conference, ICARIS 2011, Cambridge, UK, Lecture Notes in Computer Science 6825 Springer 2011*, pp. 348-360, 2011.
[63] C. V. Rao, "An Introduction to Immunology," *CRC Press LLC*, vol. ISBN: 0849313201, 2001.
[64] L. N. d. Castro and F. J. V. Zuben, "Artificial Immune Systems: Part I - Basic Theory and Applications," *Technical Report - RT DCA 01/99* *http://eva.evannai.inf.uc3m.es/docencia/doctorado/cib/documentacion/OverviewIS.pdf*, 1999.
[65] "How Vaccines Work," *NPI Reference Guide on Vaccines and Vaccine Safety* *www.path.org/vaccineresources/files/How_Vaccines_Work.pdf*, pp. 5-8.
[66] P. Matzinger, "Tolerance, danger and the extended family," *Annual Review of Immunology*, vol. 12, pp. 991-1045, 1994.
[67] L. N. d. Castro and J. Timmis, "Artififial immune system: a new computational intelligence approach," *Springer*, 2002.
[68] S. Forrest, A. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," *Proceedings - IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 202-212, 1994.
[69] T. Stibor, P. Mohr, J. Timmis, and C. Eckert, "Is Negative Selection Appropriate for Anomaly Detection?," *In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 321-328, 2005.

[70] T. Stibor, J. Timmis, and C. Eckert, "A Comparative Study of Real-Valued Negative Selection to Statistical Anomaly Detection Techniques," *In Proceedings of the 4th International Conference on Artificial Immune Systems (ICARIS)*, vol. LNCS: 3627, pp. 262-275, 2005.

[71] T. Stibor, J. Timmis, and C. Eckert, "On the Appropriateness of Negative Selection defined over Hamming Shape-Space as a Network Intrusion Detection System," *Proceedings of the Congress on Evolutionary Computation (CEC)*, 2005.

[72] T. Stibor, J. Timmis, and C. Eckert, "On Permutation Masks in Hamming Negative Selection," *Proceedings of the 5th International Conference on Artificial Immune Systems (ICARIS)*, vol. LNCS: 4163 pp. 122-135, 2006.

[73] T. Stibor, J. Timmis, and C. Eckert, "Generalization Regions in Hamming Negative Selection," *Intelligent Information Systems (IIS), Advances in Soft Computing*, pp. 447-456, 2006.

[74] L. N. d. Castro and J. Zuben, "The Clonal Selection Algorithm with Engineering Applications," *Workshop Proceedings of GECCO, Workshop on Artificial Immune Systems and Their Applications, Las Vegas*, pp. 36-37, 2000.

[75] J. Brownlee, "Clonal Selection Theory and CLONALG: The Clonal Selection Classification Algorithm (CSCA)," *Technical Report 2-02, CISCP, Swinburne University of Technology.*, 2005.

[76] J. Timmis, A. Hone, T. Stibor, and E. Clark, "Theoretical advances in artificial immune systems," *Theoretical Computer Science*, vol. 403, pp. 11-32, 2008.

[77] E. Clark, A. Hone, and J. Timmis, "Artificial Immune Systems A Markov Chain Model of the B-Cell Algorithm," *4th International Conference, ICARIS, LNCS: 3627*, pp. 318-330, 2005.

[78] N. K. Jerne, "THE GENERATIVE GRAMMAR OF THE IMMUNE SYSTEM," *Nobel Lecture, 8 December 1984*, 1984.

[79] L. N. d. Castro and F. J. V. Zuben, "aiNet: An Artificial Immune Network for Data Analysis," *Data Mining: A Heuristic Approach*, vol. 1, pp. 231-260, 2002.

[80] J. Timmis and M. Neal, "A Resource Limited Artificial Immune System for Data Analysis," *Knowledge Based Systems*, vol. 14, pp. 121-130, 2001.

[81] A. Secker, A. Freitas, and J. Timmis, "A danger theory inspired approach to web mining," *Second International Conference on Artificial Immune system (ICARIS 2003) LNCS: 2787*, 2003.

[82] J. Greensmith, U. Aickelin, and J. Twycross, "Detecting Danger: Applying a Novel Immunological Concept to Intrusion Detection Systems," *Proceedings of the 6th International Conference in Adaptive Computing in Design and Manufacture (ACDM2004)*, 2004.

[83] J. Greensmith, U. Aickelin, and S. Cayzer, "Introducing Dendritic Cells as a Novel Immune-Inspired Algorithm for Anomoly Detection," *Proceedings of the 4th International Conference on Artificial Immune Systems (ICARIS2005), LNCS: 3627*, pp. 153-167, 2005.

[84] J. Greensmith, J. Twycross, and U. Aickelin, "Dendritic Cells for Anomaly Detection," *Proceedings of the IEEE Congress on Evolutionary Computation (CEC2006)*, pp. 664-671, 2006.

[85] J. Greensmith and U. Aickelin, "The Deterministic Dendritic Cell Algorithm," *Proceedings of the 7th International Conference on Artificial Immune Systems (ICARIS 2008), LNCS: 5132*, pp. 291-303, 2008.

[86] J. Kim, J. Greensmith, J. Twycross, and U. Aickelin, "Malicious Code Execution Detection and Response Immune System inspired by the Danger Theory," *Proceedings of Adaptive and Resilient Computing Security Workshop (ARCS-05)*, 2005.

[87] "AISWeb: The Online Home of Artificial Immune Systems," *Basic Immune Inspired Algorithms* *http://www.artificial-immune-systems.org/algorithms.shtml*.

[88] 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, pp. 291-317, 2004.

[89] J. Timmis, M. Neal, and J. Hunt, "An artificial immune system for data analysis " *Biosystems*, vol. 55, pp. 143-150, 2000.
[90] C. McEwan and E. Hart, "On AIRS and Clonal Selection for Machine Learning," *8th International Conference, ICARIS, LNCS: 5666*, pp. 67–79, 2009.
[91] P. K. Harmer, P. D. Williams, G. H. Gunsch, and G. B. Lamont, "An artificial immune system architecture for computer security applications," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 252 - 280 2002.
[92] M. Ayara, J. Timmis, R. d. Lemos, and S. Forrest, "Immunising Automated Teller Machines," *Jacob et al (Eds.): ICARIS 2005, LNCS 3627*, pp. 404-417, 2005.
[93] A. Khaled, H. M. Abdul-Kader, and N. A. Ismail, "Artificial Immune Clonal Selection Algorithm: A Comparative Study of CLONALG, opt-IA and BCA with Numerical Optimization Problems," *International Journal of Computer Science and Network Security*, vol. 10, pp. 24-30, 2010.
[94] A. M. Whitbrook, U. Aickelin, and J. M. Garibaldi, "Idiotypic Immune Networks in Mobile Robot Control," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: CYBERNETICS*, vol. 37, pp. 1581-1598, 2007.
[95] A. M. Whitbrook, U. Aickelin, and J. M. Garibaldi, "The transfer of evolved artificial immune system behaviours between small and large scale robotic platforms," *Proceedings of the 9th international conference on Artificial evolution (EA'09)*, 2009.
[96] A. M. Whitbrook, U. Aickelin, and J. M. Garibaldi, "An Idiotypic Immune Network as a Short-Term Learning Architecture for Mobile Robots," *7th International Conference, ICARIS, LNCS: 5132*, pp. 266 – 278, 2008.
[97] H. Lau, I. Bate, and J. Timmis, "An Immuno-engineering Approach for Anomaly Detection in Swarm Robotics," *8th International Conference, ICARIS 2009, LNCS: 5666*, pp. 136–150, 2009.
[98] L. N. d. Castro and F. J. V. Zuben, "Artificial Immune Systems: Part II - A Survey of Applications," *Technical Report - RT DCA 02/00* *ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/lnunes/rtdca0200.pdf*, 2000.
[99] J. Kim and P. Bentley, "Immune Memory in the Dynamic Clonal Selection Algorithm," *Proceedings of the First International Conference on Artificial Immune Systems (ICARIS2002)*, pp. 59-67, 2002.
[100] D. Dal, S. Abraham, A. Abraham, S. Sanyal, and M. sanglikar, "Evolution Induced Secondary Immunity: An Artificial Immune System Based Intrusion Detection System," *Computer Information Systems and Industrial Management Applications (CISIM)*, 2008.
[101] X. Yue, A. Abraham, and Z.-X. Chi, "Artificial immune system inspired behaviour-based anti-spam filter," *Soft Computing*, vol. 11, pp. 729-740, 2007.
[102] A. Secker, A. Freitas, and J. Timmis, "AISEC: An artificial immune system for e-mail classification," *Proceedings of the Congress on Evolutionary Computation*, pp. 131--139, 2003.
[103] N. Prattipati and E. Hart, "Evaluation and Extension of the AISEC Email Classification System," *7th International Conference ICARIS, LNCS: 5132*, pp. 154–165, 2008.
[104] D. Dasgupta, S. Yu, and N. S. Majumdar, "MILA - multilevel immune learning algorithm and its application to anomaly detection," *Soft Computing*, 2003.
[105] Z. Ji and D. Dasgupta, "Real-valued negative selection algorithm with variable-sized detectors," *Proc. of GECCO, Springer, LNCS: 3102*, pp. 287-298, 2004.
[106] D. Dasgupta, K. Krishnakumar, D. Wong, and M. Berry, "Immunity-based aircraft fault detection system," *1st Intelligent Systems Technical Conference*, 2004.
[107] L. Shulin, Z. Jiazhong, S. Wengang, and H. Wenhu, "Negative-selection algorithm based approach for fault diagnosis of rotary machinary," *Proceedings of the American Control Conference*, pp. 3955-3960, 2002.
[108] R. Schulze, F. Dietel, J. Jakel, and H. Richter, "Using an artificial immune system for classifying aerodynamic instabilities of centrifugal compressors," *Congress on Nature and Biological Inspired Computing (NaBIC)*, pp. 31-36, 2010.

[109] R. L. Fanelli, "A Hybrid Model for Immune Inspired Network Intrusion Detection," *7th International Conference ICARIS, LNCS: 5132,* pp. 107–118, 2008.
[110] J. Zeng and J. Zeng, "A Novel Immunity-Based Anomaly Detection Method," *International Seminar on Future BioMedical Information Engineering,* pp. 195-198, 2008.
[111] W. Ma, D. Tran, and D. Sharma, "Negative Selection with Antigen Feedback in Intrusion Detection," *7th International Conference, ICARIS, LNCS: 5132,* pp. 200–209, 2008.
[112] Z. Ji and D. Dasgupta, "Revisiting negative selection algorithms," *Evolutionary Computation,* vol. 15, pp. 223-251, 2007.
[113] L. N. d. Castro and J. Timmis, "An Artificial Immune Network for Multimodal Function Optimisation," *Proceedings Of IEEE World Congress on Evolutionary Computation,* pp. 669-674, 2002.
[114] J. Kelsey and J. Timmis, "Immune inspired somatic contiguous hypermutation for function optimisation," *Genetic and Evolutionary Computation Conference - GECCO,* pp. 207–218, 2003.
[115] V. Cutello, G. Nicosia, and M. Pavone, "Exploring the capability of immune algorithms: A characterization of hypermutation operators," *Proceedings of the Third International Conference on Artificial Immune System (ICARIS'04)* pp. 263-276, 2004.
[116] K. A. Al-Sheshtawi, H. M. Abdul-Kadir, and A. A. Ismail, "Artificial Immune Clonal Selection Algorithms: A Comparative Study of CLONALG, opt-IA and BCA with Numerical Optimization Problems," *International Journal of Computer Science and Network Security (IJCSNS),* vol. 10, pp. 24-30, 2010.
[117] S. Xuhua and Q. Feng, "An optimization Algorithm Based on Multi-population Artificial Immune Network," *Fifth International Conference on Natural Computation,* pp. 379-383, 2009.
[118] V. S. Aragon, S. C. Esquivel, and C. A. C. Coello, "Artificial Immune System for Solving Constrained Optimization Problems," *Inteligencia Artificial,* pp. 55-66, 2007.
[119] S. Wang and X. Xu, "A novel immune clonal selection optimization algorithm," *International Conference on Computer Application and System Modeling (ICCASM'10),* pp. 391-395, 2010.
[120] W. F. El-Wahed, E. M. Zaki, and A. M. El-Refaey, "Reference Point Based Multi-Objective Optimization Using Hybrid Artificial Immune System," *Universal Journal of Computer Science and Engineering Technology,* vol. 1, pp. 24-30, 2010.
[121] P. A. D. Castro and F. J. V. Zuben, "BAIS: A Bayesian Artificial Immune System for the Effective Handling of Building Blocks," *Information Sciences* vol. 179, pp. 1426-1440, 2009.
[122] P. A. D. Castro and F. J. V. Zuben, "MOBAIS: A Bayesian Artificial Immune System for Multi-Objective Optimization," *7th International Conference, ICARIS, LNCS: 5132,* pp. 48-59, 2008.
[123] V. S. Aragon, S. C. Esquivel, and C. A. C. Coello, "Artificial Immune System for Solving Global Optimization Problems," *Inteligencia Artificial,* vol. 46, pp. 3-16, 2010.
[124] V. S. Aragon, S. C. Esquivel, and C. A. C. Coello, "A T-cell Algorithm for Solving Dynamic Optimization Problems," *Information Sciences,* vol. 181, pp. 3614-3637, 2011.
[125] J. Qing, X. Liang, R. Bie, and X. Gao, "A New Clustering Algorithm Based on Artificial Immune Network and K-means Method," *Sixth International Conference on Natural Computation,* pp. 2826-2830, 2010 2010.
[126] R. Younsi and W. Wang, "A New Artificial Immune System Algorithm for Clustering," *Lecture Notes in Computer Science,* pp. 58-64, 2004.
[127] T. Liu, Y. Zhou, and Z. Hu, "A New Clustering Algorithm Based on Artificial Immune System," *Fifth International Conference on Fuzzy Systems and Knowledge Discovery,* pp. 347-351, 2008.

[128] J. Timmis and T. Knight, "An Immmunological Approach to Data Mining," *Proceedings (IEEE International Conference on Data Mining),* vol. 1, pp. 297-304, 2001.
[129] O. Nasraoui, F. Gonzalez, C. Cardona, C. Rojas, and D. Dasgupta, "A Scaleable Artificial Immune System Model for Dynamic Unsupervised Learning," *Genetic and Evolutionary Computation Conference (GECCO), LNCS: 2723,* pp. 219-230, 2003.
[130] X. Li, T. Lu, Z. Wang, and C. Gao, "ICAIS: A Novel Incremental Clustering Algorithm Based on Artificial Immune System," *International Conference on Internet Computing in Science and Engineering,* pp. 85-90, 2008.
[131] Z. Liu, X. Jin, R. Bie, and X. Gao, "FAISC: a Fuzzy Artificial Immune System Clustering Algorithm," *Third International Conference on NAtural Computation (ICNC 2007),* 2007.
[132] M. Rabbani and H. Panahi, "An Efficient Hybrid Artificial Immune Algorithm for Clustering," *Eighth International Conference on Hybrid Intelligent Systems,* pp. 374-379, 2008.
[133] J.-H. Jiang, J. H. Wang, X. Chu, and R.-Q. Yu, "Clustering Data using a Modified Integer Genetic Algorithm (IGA)," *Analytica Chimica Acta,* vol. 354, 1997.
[134] F. Glover, "Tabu Search-Part I," *Operations Research Society of America,* vol. 1, pp. 190-206, 1989.
[135] C.-Y. Chiu and C.-H. Lin, "Cluster Analysis Based on Artificial Immune System and Ant Algorithm," *Third International Conference on Natural Computation (ICNC 2007),* 2007.
[136] N. C. Woolley and J. V. Milanovic, "Application of AIS Based Classification Algorithms to Detect Overloaded Areas in Power System Networks," *8th International Conference, ICARIS 2009, LNCS: 5666,* pp. 165–177, 2009.
[137] F. H. Arnold, "Design by Directed Evolution," *Accounts of chemical research,* vol. 31, p. 125, 1998.
[138] C. Fraley, "Algorithms for model-based Gaussian hierarchical clustering," *SIAM Journal on Scientific Computing,* vol. 20, p. 270, 1999.
[139] M. Tavallaee, E. Bagheri, L. Wei, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications CISDA* 2009.
[140] L. Duan, L. Xu, Y. Liu, and J. Lee, "Cluster-based outlier detection," *Annals of Operations Research,* vol. 168, pp. 151-168, 2009.
[141] M. B. A.-. Zoubi, "An Effective Clustering-Based Approach for Outlier Detection," *European Journal of Scientific Research,* vol. 28, pp. 310-316, 2009.
[142] J. Zhang and H. Wang, "Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance," *Knowledge and Information Systems,* vol. 10, pp. 333-355, 2006.
[143] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *Proceedings of ACM SIGMOD Conference, Montreal, Canada,,* pp. 103-114, 1996.
[144] T. N. Raymond and J. Han, "CLARANS: a method for clustering objects for spatial data mining," *IEEE Transactions on Knowledge and Data Engineering,* vol. 14, pp. 1003-1016, 2002.
[145] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noises," *2nd Int. Conferance on Knowledge Discovery and Data Mining (KDD-96),* pp. 226-231, 1996.
[146] E. M. Knorr and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," *Proceedings of the 24rd International Conference on Very Large Data Bases,* pp. 392–403, 1998.
[147] F. Angiulli and C. Pizzuti, "Outlier Mining in Large High-Dimensional Data Sets," *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, pp. 203-215, 2005.
[148] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," *SIGMOD record,* vol. 29, pp. 427-438, 2000.

[149] V. Barnett and T. Lewis, "Outliers in Statistical Data," *New York: Wiley*, 1994.
[150] J. W. Tukey, "Exploratory data analysis," *Reading: Addison–Wesley*, 1977.
[151] P. Rousseeuw and A. Leroy, "Robust Regression and Outlier Detection," *3rd ed. John Wiley & Sons*, 1996.
[152] S. Papadimitriou, H. Kitawaga, P. Gibbons, and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral," *Proc. of the International Conference on Data Engineering*, pp. 315-326, 2003.
[153] P. Yang and B. Huang, "A Spatial Clustering Algorithm for Outlier Detection," *Int. Seminar on Future Information Technology and management Engineering*, pp. 33-36, 2008.
[154] M. Breunig, H. Kriegel, R. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," *Proc. ACM SIGMOD Int. Conf. On Management of Data*, pp. 93–104, 2000.
[155] M. F. Jiang, S. S. Tseng, and C. M. Su, "Two-phase clustering process for outliers detection," *Pattern Recognition Letters*, vol. 22, pp. 691–700, 2001.
[156] J. Almeida, L. Barbosa, A. Pais, and S. Formosinho, "Improving Hierarchical Cluster Analysis: A New Method with Outlier Detection and Automatic Clustering," *Chemometrics and Intelligent Laboratory Systems*, vol. 87, pp. 208–217, 2007.
[157] A. Loureiro, L. Torgo, and C. Soares, "Outlier Detection using Clustering Methods: a Data Cleaning Application," *Proceedings of KDNet Symposium on Knowledge-based Systems for the Public Sector. Bonn, Germany*, 2004.
[158] L. Kaufman and P. J. Rousseeuw, "Clustering by means of Medoids," *Statistical Data Analysis Based on the L1–Norm and Related Methods*, vol. edited by Y. Dodge, North-Holland, pp. 405-416, 1987.
[159] E. Acuna and C. Rodriguez, "A Meta Analysis Study of Outlier Detection Methods in Classification," *Technical paper, Department of Mathematics, University of Puerto Rico at Mayaguez, available at academic.uprm.edu/~eacuna/paperout.pdf. In proceedings IPSI, Venice*, 2004.
[160] "Department of Health and Chief Medical Officer. Good doctors, safer patients - Proposal to strengthen the system to assure and improve the performance of doctors and to protect the safety of patients," *London: Department of Health*, 2006.
[161] "Department of Health - Workforce Directorate. Medical Revalidation - Principles and Next Steps," 2008.
[162] "UK Parliament Secretary of State for Health. Trust, Assurance and Safety – The Regulations of Health Professionals in the 21st Century," *Cm 7013*, 2007.
[163] L. Baker, M. Greco, and A. Narayanan, "Doctors using patient feedback to establish professional learning goals: Results from a communication skill development program," *In Wayne Pease, Malcolm Cooper and Raj Garurajan (Eds.) Biomedical Knowledge Management: Infrastructures and Processes for eHealth Systems, IGI Global, Chapter 22*, pp. 303-314, 2010.
[164] J. Campbel, A. Narayanan, B. Burford, and M. Greco, "Validation of a multi-source feedback tool for use in General Practice," *Education for Primary Care*, vol. 21, pp. 165-179, 2010.
[165] J. Campbell, S. Richards, A. Dickens, M. Greco, A. Narayanan, and S. Brearley, "Assessing the professional performance of UK doctors: an evaluation of the utility of the General Medical Council patient and colleague questionnaires," *Qual. Saf. Health Care*, vol. 17, pp. 93-187, 2008.
[166] A. Narayanan and M. Greco, "What distinguishes general practitioners from consultants, according to patients?," *Journal of Healthcare Management and Marketing*, vol. 1, pp. 80-87, 2007.
[167] A. Narayanan, M. Greco, and J. Campbell, "Generalisability for unbalanced, uncrossed and fully nested studies," *Medical Education*, vol. 44, pp. 367-378, 2010.
[168] M. A. Potter and K. A. DeJong, "The co-evolution of antibodies for concept learning," *Fifth International Conference on Parallel Problem Solving From Nature*, pp. 530-539, 1998.

[169] M. R. Ahmadi and D. Maleki, "A co-evolutionary immune system framework in a grid environment for enterprise network security," *SSI*, pp. 1136-1143, 2006.
[170] P. Hajela, J. Yoo, and J. Lee, "GA BASED SIMULATION OF IMMUNE NETWORKS APPLICATIONS IN STRUCTURAL OPTIMIZATION," *Engineering Optimization*, vol. 29, pp. 131-149, 1997.
[171] S. J. Louis and J. McDonnell, "Learning with case-injected genetic algorithms," *IEEE Transactions on Evolutionary Computation* vol. 8, pp. 316-328, 2004.
[172] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On Clustering Validation Techniques," *Journal of Intelligent Information Systems*, vol. 17, pp. 107-145, 2001.
[173] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: part I," *SIGMOD record*, vol. 31, pp. 40-45, 2002.
[174] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: part II," *SIGMOD record*, vol. 31, pp. 19-27, 2002.
[175] J. C. Bezdek and N. R. Pal, "Some New Indexes of Cluster Validity," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: CYBERNETICS*, vol. 28, pp. 301-315, 1998.
[176] J. M. Benyus, "Biomimicry: Innovation Inspired by Nature," *Perennial, New York*, 2002.
[177] C. Fraley and A. Raftery, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis," *Computer Journal*, vol. 41, pp. 578-588, 1998.
[178] W. L. Martinez and A. Martinez, "Model-based Clustering Toolbox for MATLAB," *Interface*, 2005.
[179] S. Carroll, "From Eternity to Here: The Quest for the Ultimate Theory of Time," *Dutton*, vol. ISBN 0525951334, 2010.
[180] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica (Ljubljana)*, vol. 31, pp. 249-268, 2007.
[181] J. R. Quinlan, "Discovering rules by induction from large collection of examples," *D. Michie (ed.), Expert Systems in the Microelectronic Age*, pp. 168-201, 1979.
[182] W. Cohen, "Fast, Effective Rule Induction," *In Proceedings of ICML-95*, pp. 115-123, 1995.
[183] J. Furnkranz, "Separate-and-Conquer Rule Learning," *Artificial Intelligence Review*, vol. 13, pp. 3-54, 1999.
[184] J. Robert and L. C. J. Howlett, "Radial Basis Function Networks 2: New Addvances in Design," 2001.
[185] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *In: Rumelhart D E, McClelland J L et al. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, MA, Vol. 1*, pp. 318-362, 1986.
[186] G. Zhang, "Neural networks for classification: a survey " *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 30, pp. 451-462, 2000.
[187] D. Heckerman, C. Meek, and G. Cooper, "A Bayesian Approach to Causal Discovery," *In Glymour, C. and G. Cooper, (ed.), Computation, Causation, and Discovery, MIT Press*, pp. 141-165, 1999.
[188] D. Aha, "Lazy Learning," *Dordrecht: Kluwer Academic Publishers*, 1997.
[189] V. Vapnik, "The Nature of Statistical Learning Theory," *Springer Verlag* vol. ISBN 0-387-9878-0, 1995.
[190] S. Wierzchon and U. Kuzelewska, "Stable Clusters Formation in an Artificial Immune System," *In Timmis, J., Bentley, P.J., eds.: Proceedings of the International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, pp. 68-75, 2002.
[191] A. Sharma and D. Sharma, "Clonal Selection Algorithm for Classification," *ICARIS: 2011, LNCS 6825*, pp. 361-370, 2011.
[192] I. Fischer, F. Hennecke, C. Bannes, and A. Zell, "Java Neural Network Simulator - JavaNNS," *University of Tubingen* http://www.ra.cs.uni-tuebingen.de/software/JavaNNS/.
[193] "UCI Machine Learning Repository http://archive.ics.uci.edu/ml/ "

[194] G. R. Iversen and H. Norpoth, "Analysis of Variance," *Sage University Papers Series on Quantitative Applications in the Social Sciences, series no. 07-001*, 1987.
[195] W. Zhu, N. Zeng, and N. Wang, "Sensitivity, Specificity, Accuracy, Associated Confidence Interval and ROC Analysis with Practical SAS Implementations" *Health Care and Life Sciences, NESUG (2010)*.
[196] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, pp. 37-46, 1960.
[197] A. Watkins, "Exploiting Immunological Metaphors in the Development of Serial, Parallel, and Distributed Learning Algorithms," *PhD dissertation, University of Kent, Canterbury, UK*, *http://www.cse.msstate.edu/~andrew/research/publications.html*, 2005.
[198] S. M. Garrett, "How do we evaluate artificial immune systems?," *Evolutionary Computation.*, vol. 13, pp. 145-178, 2005.
[199] U. Respository, "Breast Cancer Wisconsin (Original) Data," *http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29*.
[200] "UCI Repository, Heart, http://archive.ics.uci.edu/ml/datasets/Statlog+%28Heart%29."
[201] R. Chelouah and P. Siarry, "Tabu Search applied to global optimization," *European Journal of Operational Research*, vol. 123, pp. 256-270, 2000.

Appendix A

A brief description of the real-world datasets used in this thesis is given in this appendix. For further information regarding these datasets, please go to the UCI machine learning data repository [193].

A.1 Iris Data

This is the most commonly used dataset in machine learning literature. The Iris data is a multivariate dataset and has three classes, with each class consisting of 50 instances. The properties of the Iris data are [193] as follows:

Attributes:

Sepal Length in cm
Sepal width in cm
Petal length in cm
Petal width in cm

Classes:

Iris Setosa
Iris Versicolor
Iris Virginica

A.2 Breast Cancer Wisconsin (Diagnostic) Data

This dataset has 569 instances and 32 Features including a class label. All 31 features are real-valued data features. This dataset has two classes labeled ‘Malignant’ and ‘Benign’. Class distributions are 357 Benign and 212 malignant.

A.3 Boston Data

This dataset contains 506 instances and 14 features.

A.4 Wine Data

The Wine data has 13 features and 3 clusters with a total of 178 instances. All three clusters vary in size. The attributes of this dataset consists of the results of a chemical analysis of wine grown in same region in Italy. The attributes are alcohol, malic acid,

ash, alkalinity of ash, magnesium, total phenols, flavanoids, non-flavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 and proline.

A.5 Thyroids Data

This dataset contains 215 instances, five attributes and three classes.

A.6 Breast Cancer Wisconsin (Original) Data

This dataset has 699 instances with 16 missing values, and these instances are removed. It has 10 features including class labels ('Benign' and 'Malignant'). Samples arrive periodically as Dr. Wolberg reports his clinical cases. The database therefore reflects this chronological grouping of the data. The grouping information appears below [199].

Group 1: 367 instances (January 1989)
Group 2: 70 instances (October 1989)
Group 3: 31 instances (February 1990)
Group 4: 17 instances (April 1990)
Group 5: 48 instances (August 1990)
Group 6: 49 instances (updated January 1991)
Group 7: 31 instances (June 1991)
Group 8: 86 instances (November 1991)
Total: 699 points

A.7 KDD Intrusion Detection Data

This is a very large dataset. The KDD intrusion data samples for 5 classes: Normal, Denial of Service (DOS), Probe, User-to-Root and Remote-to-Local. The dataset has both continuous and discrete features. More detail about this dataset can be found in [139].

A.8 Parkinson's Disease Data

This dataset has 197 instances each with 22 real-valued attributes. The two class labels are 'Normal' and 'Disease'. This dataset is composed of a range of biomedical voice measurements. The main objective of this dataset is to discriminate healthy/normal people from those with Parkinson's disease.

A.9 Statlog (Heart) Data

This dataset has 13 features and 270 real-valued instances. The two classes are absence and presence of heart disease. The 13 attributes [200] are:

1. Age
2. Sex
3. Chest pain type (4 values)
4. Resting blood pressure
5. Serum cholestoral in mg/dl
6. Fasting blood sugar > 120 mg/dl
7. Resting electrocardiographic results (values 0,1,2)
8. Maximum heart rate achieved
9. Exercise induced angina
10. Oldpeak = ST depression induced by exercise relative to rest
11. The slope of the peak exercise ST segment
12. Number of major vessels (0-3) colored by flourosopy
13. Thal: 3 = normal; 6 = fixed defect; 7 = reversible defect

Appendix B

B.1 HAIS for Optimization

In this thesis we implemented the HAIS algorithm in the domains of supervised and unsupervised learning. The main motivation of this appendix is to extend the HAIS algorithm for numerical function optimization.

Casstro and Timmis [113] presented an optimization version of aiNet, which was originally proposed for information compression and data clustering. The main characteristics of the proposed algorithm are its ability to find and store multiple local optimal solutions. CLONALG [74] is an example of clonal selection algorithm, originally specified for binary character recognition and engineering optimization. More detail regarding AIS optimization algorithms can be found in chapter 2.

AISs are local search algorithms where hypermutation is used to explore neighboring search space. The random population of solutions are inserted during the course of the algorithm to encourage global search. One of the drawbacks of introducing a randomly generated population of B-cells is that the algorithm can tend to repeat local searches due to the presence of a basis of attraction. Basis of attraction can be defined as follows: '*Given any local optima, consider the set of all points in the search space that have the property that starting local search from there, finishes at that local optima*.'

Most of the approaches in the literature do not address this problem, with the exception of 'tabu search' [201]. Tabu search does this by keeping a memory (the tabu list) of recent moves, which are disallowed for a certain period.

The generation and presence of memory cells is a salient feature of an NIS. In the context of AIS approaches, memory cells are points in a search space that have been visited in the past and there is no need to revisit them. Existing AIS optimization approaches do not utilize this feature. The salient feature of our proposed HAIS-optimization in comparison with existing AIS optimization algorithms is its more effective and novel role for memory cells.

Our proposed algorithm (HAIS-optimization) keeps track of all the points visited and when new B-cells are generated a negative selection is performed against the existing memory cells. The proposed HAIS-optimization algorithm is described below.

B.2 HAIS-Optimization Algorithm

Select parameters: **NegT, N, g, n, Mut**

NegT – negative clonal selection N – B-cell population size g – most affine antibodies selected n – number of antibodies generated Mut – mutation rate

a) Randomly select **'N'** number of B-cells
b) Generate Memory cells pool, and M cells := B-cells
c) **Repeat:**
d) **For each B-cell**
e) Generate **'n'** antibodies based on **'Mut'** mutation rate
f) Select antibodies with higher fitness than parent B-cell
g) Select antibody with highest affinity **(b-Ab)**
h) **If** not empty (antibodies)
    i) Perform negative selection between antibodies
    ii) Perform negative selection between M-cells and selected antibodies
    iii) Select maximum of **'g'** antibodies
    iv) **If** not empty (antibodies)
        (a) Evolve parent B-cells towards **b-Ab**
        (b) Generate new B-cells for each of the antibody
        (c) M cells = M cells + new B-cells
    v) Else
        (a) **If** fitness of **b-Ab** is better than B-cell
            1. Evolve parent B-cell towards **b-Ab**
            2. Evolve related M cell receptors towards **b-Ab**
        (b) Else
            1. Remove B-cell
i) Else
    i) Remove B-cell
j) **End For loop**
k) **If** size of B-cell population is less than **'N'**
        (a) Generate new B-cells (Add them into the B-cell population after performing negative selection with M cells)
l) **Until termination condition**

B.3 Algorithm Explanation

The algorithm starts by generating a random number of B-cells ('N'). These B-cells are also placed into the pool of memory cells (M-cells). Each of the B-cells in the system generates 'n' number of antibodies using a pre-defined mutation rate. Now, all antibodies with affinity (fitness) greater than the parent B-cell are selected. The antibody with highest affinity is kept separated and called b-Ab. At this step (Step h), if the antibody pool is not empty, we perform a negative selection. A negative selection is first performed between newly generated antibodies and subsequently between antibodies and existing M-cells. After performing a negative selection, if the antibody

pool is not empty then we perform three tasks: (a) evolve the parent B-cell receptor towards the b-Ab; (b) generate new B-cells for each of the antibodies left in the pool; and (c) assign all these newly generated B-cells to M-cells. However, if the antibody pool is empty and the fitness of the b-Ab is higher than that of the parent B-cell, the B-cell and M-cell receptors are evolved towards the b-Ab. Otherwise the parent B-cell is removed from the population of B-cells. At the end of each generation, the size of the population of B-cells is calculated and if it is below 'N', new B-cells are generated. However, before introducing new B-cells into the population of existing B-cells, a negative selection (against existing M-cells) is performed.

The outcome of this algorithm is in the form of a set of evolved M-cells. These M-cells are points explored in a search space and subsequently a set of best memory cells can be selected from them.

B.4 Experimental Results

Following are the two functions used to evaluate the performance of our proposed HAIS-optimization algorithm.

$$f = 10.2 + (x^2 - 10\cos(2\pi x)) + (y^2 - 10\cos(2\pi y)) \quad [-2,\ 2] \qquad (1)$$

$$f = \sin(x^2 + y^2) \quad [-2,\ 2] \qquad (2)$$

The parameters used for these experiments are as follows:

Gen = 25 % no. of generations
N = 20 % B-cell population size
g = 10 % most affine antibodies selected
n = 100 % no. of antibodies generated per B-cell
NegT = 0.15 % negative selection threshold
Mut = 12.0 % mutation rate
peaks = 16 % how many best M-cells are required

In Figure B-1 below, the black and red marks are the M-cells obtained by the HAIS algorithm. The red peaks are 16 best memory cells obtained from the pool of all obtained memory cells.

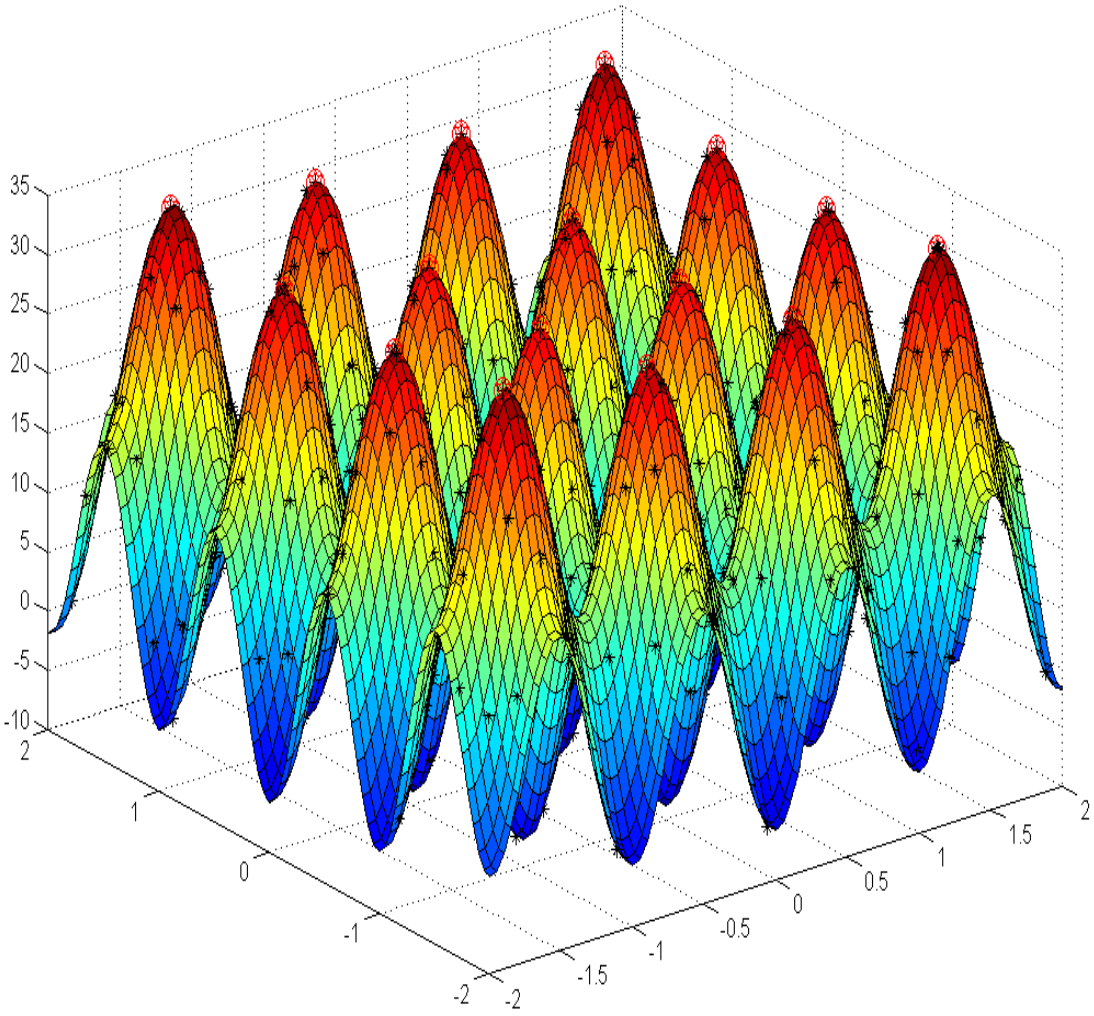


Figure B-1: M-cells obtained using the HAIS algorithm on Equation 1

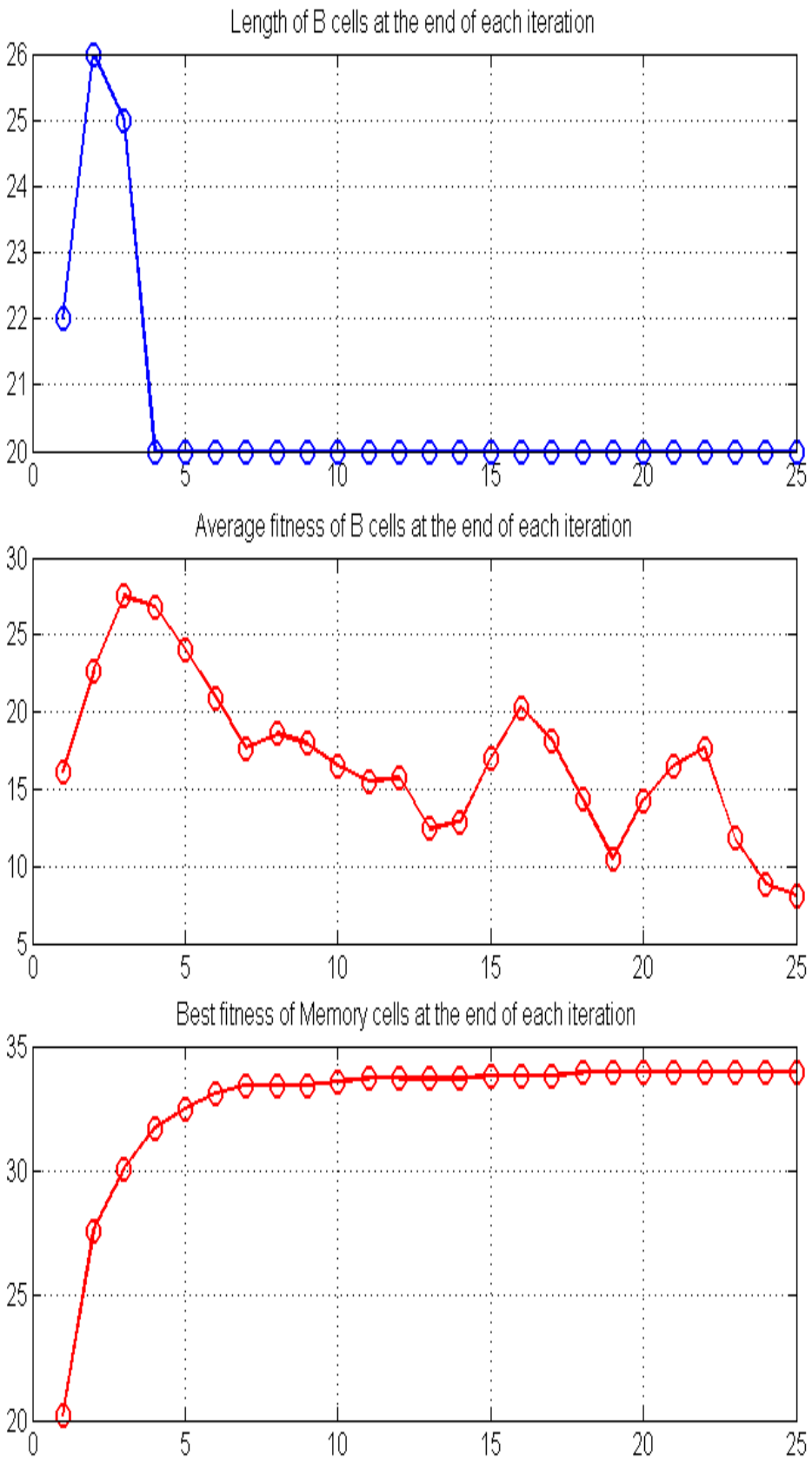


Figure B-2: Size of B-cell population, average fitness and best fitness of M-cells obtained at the end of each generation

The original 16 points, as well as the corresponding fitness values obtained by the algorithm are shown in Table B-1.

Table B-1: List of obtained best M-cells

x	y	f
1.5041	-1.5022	34.7145
1.5012	1.5054	34.7138
-1.5195	-1.5054	34.6942
-1.4863	1.5575	34.1533
1.5068	0.501	32.7121
-1.5078	0.4999	32.7114
0.5008	-1.5025	32.7069
0.5041	1.5139	32.7047
1.5084	-0.496	32.7042
-1.5064	-0.4944	32.6993
-0.5103	1.5125	32.6961
-0.5184	-1.5048	32.6617
0.4969	0.5007	30.6956
-0.5028	-0.5129	30.6813
-0.5023	0.4908	30.6752
0.4888	-0.4887	30.6281

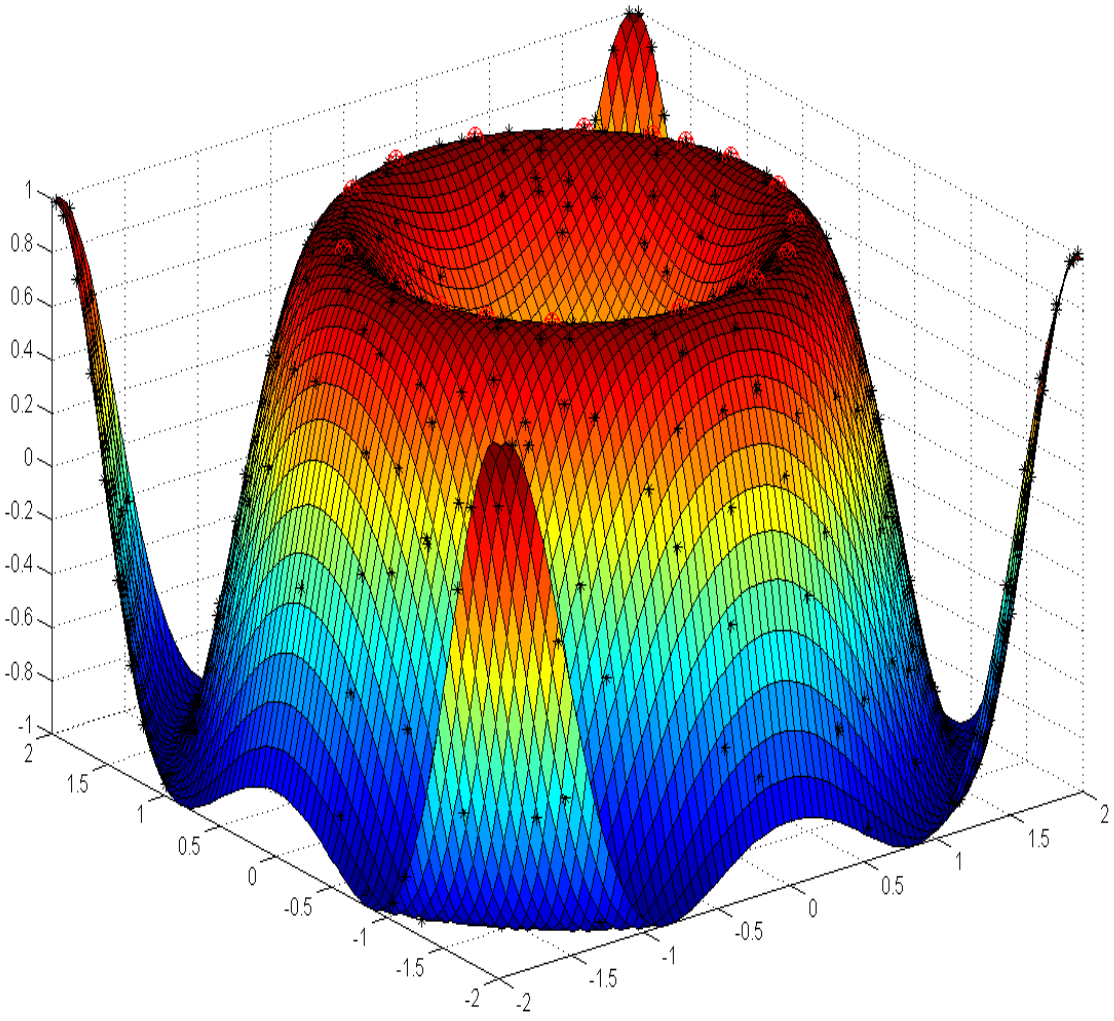


Figure B-3: M-cells obtained using the HAIS algorithm on Equation 2

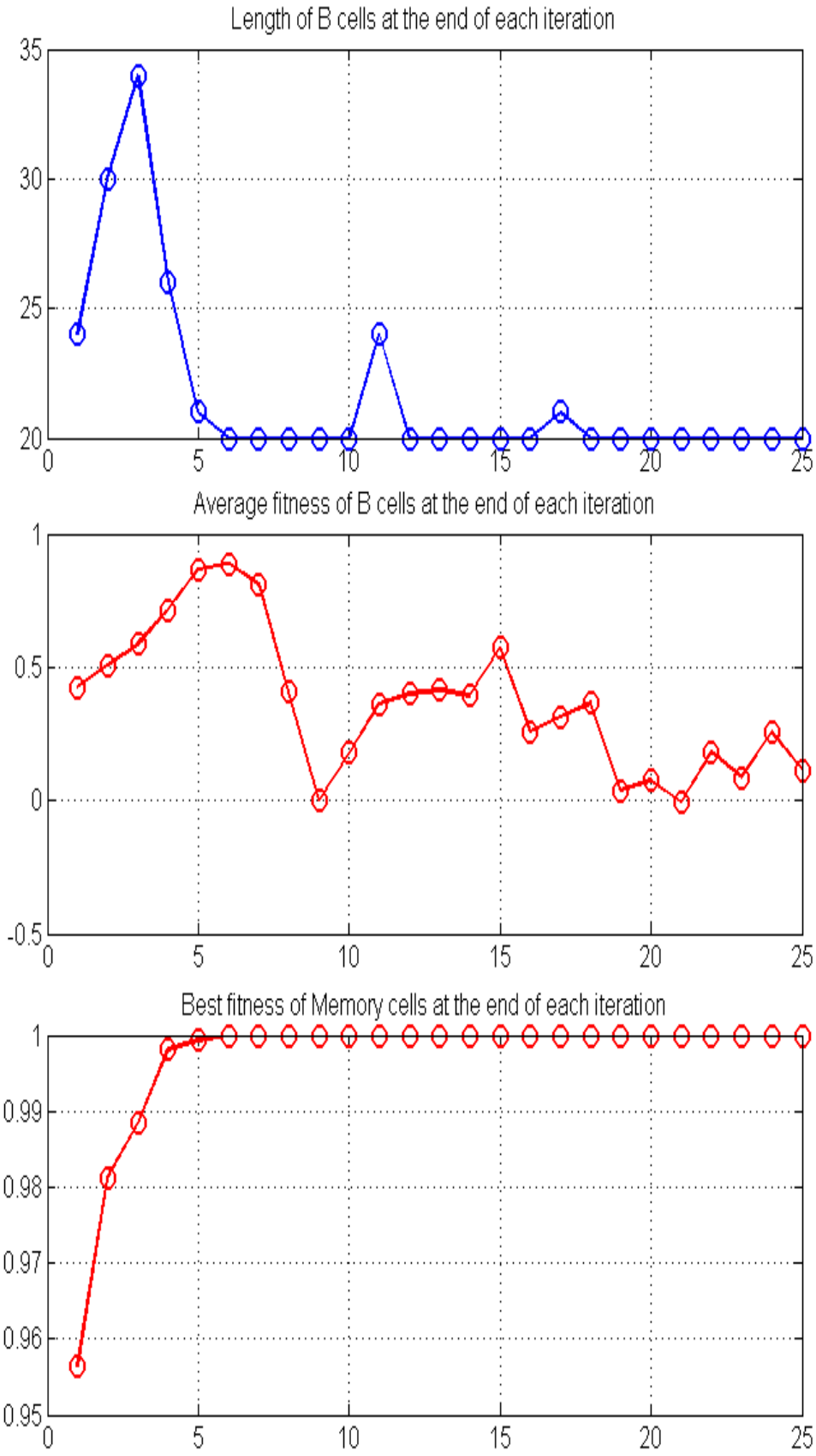


Figure B-4: Size of B-cell population, average fitness and best fitness of M-cells obtained at the end of each generation

Table B-2: List of obtained best M-cells

x	y	f
-1.1845	-0.4096	1
1.0449	-0.6921	1
0.8333	0.9362	1
-1.0623	-0.6651	1
-0.2284	1.2324	1
1.1659	0.4594	1
0.7325	-1.0167	1
-0.8247	-0.9434	1
1.2178	-0.2949	1
-1.1534	0.4913	1
-0.6507	1.0706	1
1.2435	0.1602	1
-0.1789	-1.2399	1
0.3037	1.2152	1
0.3849	-1.1938	1
1.0775	0.6379	1

Here, we have demonstrated with simple two numeric function optimization functions that the HAIS algorithm can also be fine-tuned to solve optimization problems. In the HAIS-optimization algorithm, the population of B-cells is evolved towards local optimum solutions and randomly generated B-cells are introduced into the existing population of B-cells when the total size of the population decreases to a pre-defined threshold.

Appendix C

The data partitioning results in terms of clustering errors against the true class labels obtained on three datasets using hierarchical Euclidean distance measure clustering, k-means clustering and hierarchical model-based clustering algorithms are shown in Table C-1. The features of all three datasets are normalized between -1 and +1.

Table C-1: Clustering errors obtained using various clustering algorithms

	Hierarchical Euclidean Distance Clustering	**k-means Clustering**	**Hierarchical Model-Based Clustering**
Iris	51	17	18
Wine	109	9	7
Diabetes	69	41	36