Revisiting the Foundations of Artificial Immune Systems: A Problem-oriented Perspective

Alex A. Freitas and Jon Timmis

Computing Laboratory
University of Kent
Canterbury, CT2 7NF, UK
{A.A.Freitas, J.Timmis}@kent.ac.uk

Abstract. Since their development, AIS have been used for a number of machine learning tasks including that of classification. Within the literature, there appears to be a lack of appreciation for the possible bias in the selection of various representations and affinity measures that may be introduced when employing AIS in classification tasks. Problems are then compounded when inductive bias of algorithms are not taken into account when applying seemingly generic AIS algorithms to specific application domains. This paper is an attempt at highlighting some of these issues. Using the example of classification, this paper explains the potential pitfalls in representation selection and the use of various affinity measures. Additionally, attention is given to the use of negative selection in classification and it is argued that this may be not an appropriate algorithm for such a task. This paper then presents ideas on avoiding unnecessary mistakes in the choice and design of AIS algorithms and ultimately delivered solutions.

1 Introduction

Artificial Immune Systems (AIS) are a relatively new computational intelligence paradigm [5]. As in other computational intelligence paradigms, the main goal is to design effective problem-solving algorithms, rather than to model a biological phenomenon. Intuitively the design of an AIS algorithm should be strongly determined by the kind of problem that the algorithm will try to solve. However, this is not always the case in the AIS literature. In some cases we can observe a certain mismatch between the design of the algorithm and the problem being solved by the algorithm. This paper illustrates a number of these mismatches and suggests ways to remove them, and a problem-oriented perspective is advocated for designing and applying AIS algorithms.

It should be noted that AIS algorithms are normally designed to be generic algorithms. Hence, the criticism presented in this paper is not intended to be a criticism to the design of current generic AIS algorithms nor as a criticism of the AIS paradigm as a whole. Rather, the core idea of our criticism is that, when applying a generic AIS algorithm to a well-defined and specific problem, the algorithm typically needs to be adapted to specific characteristic of the problem at hand, a new AIS algorithm more suitable to that problem must be designed.

Before we proceed, it is necessarily to specify the scope of this paper. The field of AIS is too big to be revisited as a whole in a single conference paper. Hence, this paper focuses on one kind of application of AIS, which allows us to focus the discussion on important issues in the design of AIS algorithms for that kind of application. The selected application involves classification and other related tasks involving prediction, which is sometimes referred to as supervised learning. In the classification

task the goal is to predict the class of an example (a record, or data instance), given the values of a set of attributes – called the predictor attributes – for that example. The motivation for focusing in this task is two-fold. First, this is an important task in the context of computational intelligence, and it has been extensively studied in several fields such as machine learning [15], data mining [23] and pattern recognition. Second, this task has been less studied in the field of AIS, where only recently an AIS algorithm specifically designed for classification has been proposed [22], [21]. Hence, there is a strong need for a comprehensive discussion on important issues in the design and application of AIS algorithms for classification and related tasks.

In order to organize the sequence of ideas and arguments presented in this paper, we decided to follow the high-level structure of the framework for engineering AIS algorithms proposed by [5]. According to this framework, the design of an artificial immune system contains three basic elements, namely:

a) a representation for the components of the system – in this paper we are mainly interested in the representation of antibodies and antigens in the context of the classification task;
b) an affinity measure, which quantifies the interactions between components of the system – in this paper we are mainly interested in affinity functions measuring the similarity between an antibody and an antigen, in the context of classification;
c) an immune algorithm – in this paper we are particularly interested in analyzing the effectiveness of the negative selection algorithm for the classification task.

The remainder of this paper is organized as follows. Section 2 briefly reviews the concept of inductive bias, a key concept in classification. Section 3 discusses issues in the choice of antibody/antigen representation. Section 4 discusses issues in the choice of affinity functions. Section 5 discusses issues in the use of the negative selection algorithm for classification. Finally, section 6 summarizes the paper.

We emphasize that the issues involving the negative selection algorithm are considered only in section 5. Sections 3 and 4 are independent of any AIS algorithm.

2 A Review of the Concept of Inductive Bias

This section briefly reviews the important concept of inductive bias, which will support the discussions about issues in the choice of antibody/antigen representation and affinity functions to be presented in later sections of this paper.

As pointed out by [12], given a set of observed facts (data instances), the number of hypotheses – e.g. classification rules - that imply these facts is potentially infinite. Hence, a classification algorithm *must* have an *inductive bias*. An inductive bias can be defined as any (explicit or implicit) basis for favoring one hypothesis over another, other than strict consistency with the data being mined – see [14], [15]. Note that without an inductive bias a classification algorithm would be unable to prefer one hypothesis over other consistent ones. In machine learning terminology, a classification algorithm without an inductive bias would be capable of performing only the simplest kind of learning, namely rote learning.

We emphasize here a well-known fact about classification. Any bias has a *domain-dependent* effectiveness. Since every classification algorithm has a bias, the performance of a classification algorithm strongly depends on the application domain. In other

words, claims such as "classification algorithm A is better than classification algorithm B" should only be made for a given (or a few) application domain(s). This has been shown both theoretically – see [18], [17] – and empirically – see [13].

There are two major types of inductive bias, namely representation bias and preference bias [15], [7]. Representation bias is associated with the knowledge representation used by the algorithm. For instance, suppose the algorithm's knowledge representation consists of rule conditions expressed in prepositional logic – where each condition is an attribute-value pair such as *Income > 50,000* – but not in first-order logic. Then the algorithm will not be able to discover rule conditions involving relations between two attributes, such as *Income > Expenditure*.

Preference bias is associated with the evaluation function used by an algorithm to measure the quality of a candidate hypothesis. In the Instance-Based Learning (IBL) paradigm [1], also called nearest neighbours or lazy learning, preference bias is determined mainly by the distance function used to measure the distance between a pair of examples (data items). This is directly relevant for the discussion of AIS algorithms in this paper, because many AIS algorithms also have to use some kind of distance function to measure the "affinity" between an antibody and an antigen, as will be discussed in the next sections.

3 Issues in the choice of Antibody/Antigen Representation

We follow [5] in assuming the general case in which each antibody Ab (a "pattern detector") is represented by an L-dimensional vector $\mathbf{Ab} = \langle Ab_1, Ab_2, \ldots, Ab_L \rangle$, and each antigen Ag (a data item, or record, to be classified) is represented by an L-dimensional vector $\mathbf{Ag} = \langle Ag_1, Ag_2, \ldots, Ag_L \rangle$, where L is the length (i.e., the number of coordinates) of the vectors.

At least three basic kinds of representations can be used.

Binary representations – In this case the matching between an antibody and an antigen is typically based on computing the number of bits that are the same (or different, depending on whether we want to measure similarity or distance) in a pair of vectors $\langle \mathbf{Ab}, \mathbf{Ag} \rangle$.

Continuous (numeric) representations – In this case the coordinates of the antibody and antigen vectors are either real-valued or integer-valued, and the matching between an antibody and an antigen is typically based on a distance metric such as the Euclidean distance or the Manhattan distance – the differences between these distance metrics will be discussed later.

Categorical (nominal) representations – In this case the coordinates of the antibody and antigen vectors are categorical or nominal values, such as the values *female* and *male* of the attribute *Gender*. It is important to distinguish categorical representations from continuous one because in the former there is no notion of "order" between the values, unlike continuous representations. Note that in general binary representations can be considered a particular case of categorical ones.

Finally, hybrid representations are possible and *intuitively desirable* when coping with data sets having attributes of different data types. (Note that this also holds in evolutionary algorithms for data mining [7].) This point does not seem fully appreciated in the AIS literature, where sometimes a single kind of data representation is

artificially used, and as a result the data has to be somehow "adapted" to the AIS algorithm (see below), rather than adapting the algorithm to the data. Intuitively, the latter would be more natural – and probably more effective. In some cases, the approach of "adapting" the data to the algorithm will even throw away some potentially relevant data just because the algorithm cannot handle that data. For instance, [4] apply a negative selection algorithm to a multidimensional personnel data containing both categorical and numeric data. However, instead of using a hybrid categorical/numeric representation and take all the attributes into account, they simply ignore categorical attributes and work only with numeric attributes. This approach seems unnatural and, from a problem-oriented point of view, it does not seem very effective, since it throws away potentially relevant attributes.

This could be avoided by using a hybrid categorical/numeric antibody and antigen representation, with a correspondingly adapted affinity measure. In particular, an affinity measure for categorical attributes will be discussed in subsection 4.3. That measure could be used in a number of AIS algorithms that currently handle only continuous attributes, and not categorical attributes. This includes AIRS, which, although designed for classification, seems to have the limitation – in its current version [22] – of coping only with continuous attributes.

4 Issues in the Choice of Affinity Functions

4.1 Affinity Functions for Binary Antibody/Antigen Representation

When using binary representation, a natural and simple affinity function is the well-known Hamming distance (or its complement), which counts the number of bit positions with the same value (1 or 0) in the antibody and antigen being matched. Other affinity functions have also been used, in particular the r-contiguous bits rule.

An antibody Ab and an antigen Ag are said to match under the r-contiguous bits rule if Ab and Ag have the same bit value in at least r contiguous bit positions. Ideally, the use of this affinity function (or any other affinity function) should be justified taken into account the data being mined, but this is not usually the case in the literature. The r-contiguous bits rule or its variants are often used without any specific justification for this choice [3], [4], [19]. When a justification is presented, it is usually the fact that this rule is more *biologically plausible* than the Hamming distance [9], [5] (p. 70). We do not find this argument satisfactory, for two related reasons.

First, for the general case, we do not think that this particular metaphor with biology is desirable in AIS algorithms for analysing data. Why not? Because in this case the metaphor involves a *physical* (rather than *logical*) characteristic of the natural immune system. As a rule of thumb, AIS algorithms should use metaphors based on logical (rather than physical) characteristics of the natural immune system, which intuitively tend to be more generic and so more appropriate as an inspiration to design AIS algorithms for analysing data in a virtual data space. In the particular case in question, in the natural immune system a contiguous, position-dependent matching makes sense, because the lymphocytes and antigens have a contiguous genetic material in physical 3-D space. However, in data mining and machine learning the artificial lymphocytes and antigens are *virtual* entities, and the AIS algorithm does a search in

an *abstract* data space. In this kind of space it is usually more natural to consider that the attributes (or features) represented by lymphocytes and antigens are not ordered, so that the notion of contiguousness is not a natural one, of course unless this is dictated by the application area. This point is further discussed in the next paragraphs.

Second, the argument ignores the data set being mined. The choice of a particular affinity function determines a part of the inductive bias of the AIS algorithm. As mentioned in section 2, the fact that the effectiveness of an inductive bias is entirely data set-dependent is well established in the machine learning and data mining communities. So, the choice of an affinity function should be made by taking into account the data set being mined and the problem being solved. It is important to understand that r-contiguous bits rule have a *positional bias*. For instance, in [9] each lymphocyte represents a "data-path triple" describing a connection between computers, consisting of 3 values: the source IP address, the destination IP address and the service (or port) by which the computers communicate. These 3 values are represented by a single 49-bit string. Let B_{source}, B_{dest} and B_{serv} be the (sub)strings of bits used to represent the values of those 3 variables, respectively. Hence, each lymphocyte is a string obtained by concatenating those 3 (sub)strings. There are 6 different permutations of those (sub)strings that can be used to form the string representing a lymphocypte, namely: $<B_{source}, B_{dest}, B_{serv}>$, $<B_{source}, B_{serv}, B_{dest}>$, $<B_{dest}, B_{source}, B_{serv}>$, $<B_{dest}, B_{serv}, B_{source}>$, $<B_{serv}, B_{source}, B_{dest}>$, $<B_{serv}, B_{dest}, B_{source}>$. Since the r-contiguous bits rules takes the position of the bits into account, the result of the algorithm – i.e., the evolved detectors, the true positive and false positive rate, etc. – will be different for each of those 6 permutations. This is an inductive bias, since this difference in the results has nothing to do with consistency with the data. It is a side-effect of different, arbitrary choices of (sub)string permutations.

The problem is by no means restricted to this particular application domain/data set. It is much more generic. The basic problem is that in tasks related to classification and anomaly detection each detector or pattern recognizer evolved by an AIS – corresponding to a candidate solution to the underlying problem – usually represents a *set* of attributes (features), in the mathematical sense of a set, i.e., an *unordered* collection of elements without duplications. Hence, the position of attributes is irrelevant, from the point of view of the machine learning or data mining algorithm. Indeed, the vast majority of classification algorithms treat the attributes of the data as a set, and they obtain results that are independent of the order of the attributes in the file or internal data structure used by the program. Hence, they do not have the positional bias associated with the r-contiguous bits rule. This is *not* to say that we should always remove positional bias, since any bias has a domain-dependent effectiveness (section 2). Hence, the decision on whether or not to use an affinity function with a positional bias should be made by taking into account the data set being mined.

4.2 Affinity Functions for Continuous Antibody/Antigen Representation

When dealing with numeric data, the majority of the AIS literature uses the Euclidean distance (or its complement, if measuring similarity) as the affinity function, as specified in formula (1). It is interesting to note that, in general, this choice of affinity function is not justified in the literature. Presumably, authors of AIS algorithms implicitly

assume that the Euclidean distance is a "natural" or "default" distance metric. Sometimes authors mention that other distance metrics – such as the Manhattan distance, specified in formula (2), where "|x|" denotes the absolute value of x – could be used as well, but without discussing the pros and cons of these two distance metrics.

$$\text{Dist}(Ab, Ag) = \left(\sum_{i=1}^{L} (Ab_i - Ag_i)^2 \right)^{1/2} \quad (1) \qquad \text{Dist}(Ab, Ag) = \sum_{i=1}^{L} |Ab_i - Ag_i| \quad (2)$$

An exception is the AIS textbook of [5] (p. 65), where the authors make the following comment: "Although no report of the latter [Manhattan distance] has yet been found in the literature, the Manhattan distance constitutes an interesting alternative to Euclidean distance, mainly for parallel (hardware) implementation of algorithms based on the shape-space formalism."

We agree with the basic idea of this comment, but we would like to add two comments. First, the Manhattan distance tends to be computationally more efficient than the Euclidean distance even in sequential (non-parallel) implementations, since the former involves no exponentiation or square root operation. Second, we believe that it is important to go further in the analysis of the pros and cons of these two distance metrics. In addition to the issue of computational efficiency, there is an important issue of effectiveness. These two distance metrics have different *inductive biases*, and so they tend to be effective for different kinds of data set. To understand this point, let us consider the very simple example shown in Figure 1, involving a two-dimensional data set. Antigen Ag is at the origin (coordinates <0,0>) of the graph, antibody Ab_1 is at coordinates <4,4>, and antibody Ab_2 is at coordinates <6,1>. Now, which of the two antibodies is "closer" (i.e., has higher affinity to) the antigen Ag? The answer depends on the choice of distance metric. Let Dist(Ag, Ab) be the distance between antigen Ag and antibody Ab. If we use the Euclidean Distance we have:

$$\text{Dist}(Ag, Ab_1) = (4^2 + 4^2)^{1/2} = 5.66 \quad \text{and} \quad \text{Dist}(Ag, Ab_2) = (6^2 + 1^2)^{1/2} = 6.08$$

On the other hand, if we use the Manhattan distance we have:

$$\text{Dist}(Ag, Ab_1) = 4 + 4 = 8 \quad \text{and} \quad \text{Dist}(Ag, Ab_2) = 6 + 1 = 7.$$

Hence, the nearest antibody to antigen Ag is Ab_1 according to the Euclidean distance, but it is Ab_2 according to the Manhattan distance.

Why did the two distance measures lead to such a different result? Because they have different inductive biases. In particular, the Euclidean distance overemphasizes (by comparison with the Manhattan distance) large differences in the values of one or few individual attributes (coordinates). Intuitively, this makes this distance more sensitive to noisy data. That is, a single error in the value of one coordinate in the antibody or antigen vectors can be considerably amplified by the Euclidean distance formula. By contrast, the Manhattan distance tends to be more robust to noisy data, in the sense that errors in the value of one or few attributes will have relatively little impact (by comparison with the Euclidean distance) in the computation of the distance between an antibody and an antigen.

To summarize, the choice between Euclidean distance or Manhattan distance (or any other affinity function) should not be done in an arbitrary way. This is an important choice, having an influence not only in the computational efficiency but also (and

usually more importantly) in the effectiveness of the algorithm, and this choice should be done by taking into account characteristics of the data being mined.

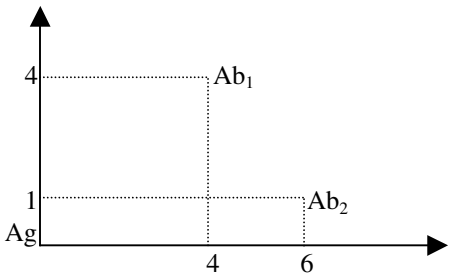

Figure 1: Euclidean distance vs. Manhattan distance – a very simple example

4.3 An Affinity Function for Categorical Antibody/Antigen Representation

The antibody/antigen representation and corresponding affinity functions of AIS algorithms usually focus on either binary or continuous (numeric) representations. From a machine learning/data mining viewpoint, this is a significant limitation, since many real-world data sets contain categorical attributes.

Hence, it is important to review here a distance measure specifically designed for coping with categorical attributes, which could be used as an affinity function in AIS algorithms where the antibodies and antigens contain categorical data. To the best of our knowledge, this measure has not been used yet in the AIS literature. This measure is called Value Difference Metric [20], [11], and it is defined by formula (3):

$$\text{Dist}(Ab_i, Ag_i) = \sum_{c \text{ in } C} (\Pr(c|Ab_i) - \Pr(c|Ag_i))^2 \quad (3)$$

where $\text{Dist}(Ab_i, Ag_i)$ is the distance between the values of the i-th attribute of the antibody Ab and antigen Ag being matched, C is the set of classes (values of the class attribute), c denotes the c-th class, and $\Pr(c|Ab_i)$ – or $\Pr(c|Ag_i)$ – denotes the empirical conditional probability that the class of an example (data instance) is c given that the example has the value Ab_i – or Ag_i – for its i-th attribute. That is, $\Pr(c|Ab_i)$ is the ratio of the number of training examples having class c and i-th attribute value Ab_i divided by the number of training examples having i-th attribute value Ab_i. $\Pr(c|Ag_i)$ is computed in the analogous way. Note that the above formula measures similarity between values of a single categorical attribute, so of course it must be applied once for each categorical attribute in the antibody/antigen representation.

The rationale for the above formula is to measure the "distance" between two categorical values as a function of the difference between the class probability distributions associated with the two values. That is why the index c in the summation symbol ranges over all classes in the set C.

5 Issues in the Use of Negative Selection for Classification

In this section we revisit the use of the negative selection algorithm in classification and related tasks, which has been a popular application of this kind of algorithm. The negative selection algorithm was originally proposed as an anomaly-detection (or change-detection) algorithm in the application domain of computer security [6]. It is based on a metaphor with the process of negative selection of T-cells in the thymus, where T-cells that match self are eliminated [5]. Hence, the mature T-cells leaving the thymus will not, in general, match self, and will therefore match only non-self.

The basic idea of the negative selection algorithm – shown at a high level of abstraction in the pseudocode of Figure 2 – is simple. The algorithm uses, as input, a set of "normal" examples, called the *self*. It iteratively generates – at random – immature T-cells and tries to match them with all the examples in the self. If a T-cell matches at least one example in the self it is discard, otherwise it is promoted to a mature T-cell and it is output by the algorithm. This iterative process is repeated until a stopping criterion is satisfied, such as sufficient coverage of the non-self space has been achieved. Once this "training phase" (originally called "censoring") is over, the set of mature T-cells are used to monitor changes or anomalies in the data in a "testing phase" (originally called "monitoring phase"). That is, each new example (data item) is compared with each mature T-cell. If the example matches a mature T-cell an anomaly or change has been detected, so that the example is considered to be a *non-self* example. Otherwise the example is considered to be a *self* example.

```
Input: a set of "normal" examples (data items), called the self (S)
Output: a set of "mature" T-cells that do not match any example in S
REPEAT
  Randomly generate an "immature" T-cell (detector)
  Measure the affinity (similarity) between this T-cell and each example in S
  IF the affinity between the T-cell and at least one example in S is greater than
     a user-defined threshold
  THEN discard this T-cell
  ELSE output this T-cell as a "mature" T-cell
UNTIL stopping criterion
```

Figure 2: Pseudocode of the Negative Selection Algorithm

In the previous discussion we have used the terms training and testing set to cast the negative selection algorithm as a kind of classification algorithm, which is often done in the AIS literature. (Note that we are *not* defending such casting, we will rather criticize it.) More precisely, the algorithm is often used to classify examples into two classes, the *self* class and the *non-self* class. In conventional machine learning terminology, non-self examples are called positive examples (since the goal of the mature T-cells is to detect these examples) and the self examples are called negative examples. Indeed, under this framework the performance measure of the algorithm typically involves the true positive (TP) rate – i.e., the number of positive (non-self) examples correctly detected by the mature T-cells divided by the total number of positive examples – and the false positive (FP) rate – i.e., the number of negative (self) examples

wrongly detected by the mature T-cells divided by the total number of negative examples. The goal is, of course, to maximize the TP rate and to minimize the FP rate. Some examples of this use of the negative selection algorithm and its variations/extensions can be found in [3], [8], [2], [10].

Let us now present a critical review of the use of the negative selection algorithm for classification and related tasks. First, the algorithm is inefficient and time-consuming, as a vast number of random detectors need to be discarded before the required number of competent detectors is obtained. Second, it should be noted that the negative selection algorithm (in its basic form) has just one means of generating detectors: at random. As pointed out by [5], p. 78, this means that the generation of detectors is not adaptive and it does not use any information in the set of self examples to guide the search. In other words, in essence the algorithm – in its basic, original form – performs a type of random search.

Furthermore, we would like to point out that the negative selection algorithm has no mechanism to minimize the danger of overfitting and oversearching – which are important mechanisms in a classification algorithm [15]. In essence, overfitting occurs when the classification algorithm learns a model that is adapted to idiosyncrasies of the training set that are unlikely to occur in an unseen test set. A related problem is oversearching [16], which occurs when the algorithm considers too many hypotheses and finds a hypothesis that reflects a spurious ("fluke") relationship, again unlikely to be true in an unseen test set.

In order to mitigate some of these limitations, the original version of the algorithm has been extended by several authors to make it adaptive. For instance, [8] proposed to use a genetic algorithm (GA) to evolve detectors in the form of IF-THEN rules covering the non-self space. [2] also proposed to use a GA as a form of affinity maturation for antibodies, although in this work the use of the GA is considered an optional aspect of the antibody life cycle, since it is very computationally expensive. Despite these advances, in general the algorithms developed in these projects are still based on one idea which is at the core of the negative selection algorithm, namely the fact that the "training" (censoring) phase of the algorithm uses only examples of one class (the self), rather than examples of two classes (self and non-self). On the other hand, the "testing" (monitoring) phase must use both self and non-self examples. After all, when monitoring new examples (say, new data packets, or new network connections), each example can be either a self (e.g., non-attack) or non-self (e.g., a network attack), and we do not know which is the true class of the example when it is being classified. This is the whole point of the classification task: the algorithm has to classify examples in the test set without knowing the true class of the example. Classification involves prediction, and the above-mentioned TP and FP rates are measures of predictive accuracy.

Now, the extended versions of the negative selection algorithms discussed in [3], [8], [2] use a test set containing both self and non-self examples – as they should, of course – but use a training set containing only self examples – due to the fact that the core of those algorithms is the negative selection algorithm, trained only on self examples, as mentioned earlier. This raises the question of how effective these algorithms are, by comparison with more conventional classification algorithms that use a training set containing data from all the classes – i.e., both self and non-self. This is an

open question at present, because unfortunately, in general comparison between these two kinds of algorithms are not reported in the literature.

To summarize, if the task being solved is classification, where we want to predict the class of examples in a test set that is completely separated from the training set, the conventional approach is to use a training set containing examples of all the classes that occur in the test set. The use of the negative selection algorithm goes against this approach, because it uses only examples of one class for training. Intuitively, this would reduce the predictive accuracy of the algorithm on the test set. Therefore, in classification or other prediction tasks the use of the negative selection algorithm does not seem to be the best approach; in principle, it would seem better to use an AIS algorithm which was specifically developed for classification, such as the AIRS algorithm [22], [21].

We emphasize that the criticism in this section refers to the use of the negative selection algorithm in classification, and *not* to the use of this algorithm in a simpler anomaly-detection task, as initially proposed by [6].

6 Summary and Future Research

In this paper we have presented an application-oriented criticism of artificial immune systems (AIS) and their use in the classification task. The motivation for this criticism is that in the AIS field this task has often been used by using generic AIS algorithms, which have not been tailored for this task. Clearly, the design of generic AIS algorithms is important, but it is also important to recognize that specific applications, such as classification, have specific requirements that have to be incorporated into the design of an AIS algorithm applied to this task.

More precisely, the classification-related issues discussed in this paper were divided into three broad groups, corresponding to the three basic elements of the AIS framework proposed by [5], namely representation, affinity function and immune algorithm.

Concerning representation, we have emphasized the importance of using hybrid antibody/antigen representations that can represent both categorical and continuous data, since both these data types are commonplace in real-world data sets. By contrast, the AIS algorithms currently being used for classification typically use either a binary or a continuous representation, and they tend to ignore categorical attributes, which limits their application.

Concerning affinity functions, most AIS algorithms for classification use functions such as Hamming distance, the r-contiguous rules and the Euclidean distance. Clearly, this choice of functions is heavily influenced by the choice of representation – binary or continuous. We pointed out that these representations have specific inductive biases that are often ignored in the AIS literature, and these biases must be considered when choosing a particular affinity function. No inductive bias is the best across all data sets. Hence, from a problem-oriented perspective, the choice of the affinity function should be made by taking into account both the inductive bias of the affinity function and the characteristics of the data being mined. We have also drawn attention to the Value Difference Metric, a distance function specifically designed for categorical attributes. This distance metric is often used in the Instance-Based Learning (IBL)

field, but it seems that it has never been used yet in the AIS field. The use of this affinity function in AIS algorithms (possibly combined with another affinity function for continuous attributes), in conjunction with the use of a categorical (or hybrid categorical/continuous) representation for antibodies and antigens, would considerably facilitate the application of AIS algorithms to data sets containing categorical attributes, which are quite common in the context of the classification task.

Concerning the immune algorithm, we have criticized the use of the negative selection algorithm (a generic AIS algorithm) in classification. More precisely, we have pointed out that the algorithm generates detectors in a random – rather than data-driven – fashion and that it has no mechanism to minimize the danger of overfitting and oversearching in the context of the classification task. The root of the problem is that the negative selection algorithm – even in its extended versions that render it more adaptive – essentially relies on a training set containing examples of a single class, whereas in classification it is important to train the algorithm in examples of all the classes that will occur in the test set.

Hence, the main contribution of this paper can be regarded as bringing concepts and principles of machine learning and data mining into the AIS field, in order to support the design of AIS algorithms that are more adapted to the classification task. In particular, among the several well-established machine learning paradigms often used in the classification task [15], the IBL paradigm seems to have a lot to offer to the AIS field, and this potential should be explored in future research.

In addition to the above example of the Value Difference Metric, another important potential contribution of the IBL paradigm to the AIS field is the use of weighted-attribute distance metrics [1]. In most classification applications, different attributes have different degrees of relevance for predicting the class attribute, which strongly suggests that AIS algorithms should be extended to use affinity functions where different attributes have different weights in the distance formula.

References

[1] D.W. Aha. (Ed.) *Artificial Intelligence Review – special issue on lazy learning, 11(1-5)*, June 1997.

[2] K.P. Anchor, P.D. Williams, G.H. Gunsch, and G.B. Lamont. The computer defense immune system: current and future research in intrusion detection. *Proc. Congress on Evolutionary Computation (CEC-2002)*. IEEE Press.

[3] J. Balthrop, F. Esponda, S. Forrest and M. Glickman. Coverage and generalization in an artificial immune system. *Proc. Genetic and Evolutionary Computation Conf. (GECCO-2002)*, pp. 3-10. Morgan Kaufmann, 2002.

[4] D. Dasgupta and N.S. Majumdar. Anomaly detection in multidimensional data using negative selection algorithm. *Proc. Congress on Evolutionary Computation (CEC-2002)*, pp. 1039-1044. IEEE Press.

[5] L.N. de Castro and J. Timmis. *Artificial Immune Systems: a new computational intelligence approach*. Springer, 2002.

[6] S. Forrest, A.S. Perelson, L. Allen and R. Cherukuri. Self-nonself discrimination in a computer. *Proc. IEEE Symp. On Research in Security and Privacy,* pp. 202-212. 1994.

[7] A.A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002.

[8] F.A. Gonzalez and D. Dasgupta. An immunogenetic technique to detect anomalies in network traffic. *Proc. Genetic and Evolutionary Computation Conf. (GECCO-2002)*, pp. 1081-1088. Morgan Kaufmann, 2002.

[9] S.A. Hofmeyr and S. Forrest. Immunity by design: an artificial immune system. *Proc. Genetic and Evolutionary Computation Conf. (GECCO-1999)*. Morgan Kaufmann, 1999.

[10] J. Kim and P.J. Bentley. Towards an artificial immune system for network intrusion detection: an investigation of dynamic clonal selection. *Proc. Congress on Evolutionary Computation (CEC-2002)*. IEEE Press.

[11] T.W. Liao, Z. Zhang, C.R. Mount. Similarity measures for retrieval in case-based reasoning systems. *Applied Artificial Intelligence, 12,* 267-288. 1998.

[12] R. W. Michalski. A theory and methodology of inductive learning. *Artificial Intelligence 20, 1983, 111-161.*

[13] D. Michie, D.J. Spiegelhalter, and C.C. Taylor. *Machine Learning, Neural and Statistical Classification.* New York: Ellis Horwood.

[14] T.M. Mitchell. The need for biases in learning generalizations. *Rutgers Technical Report, 1980*. Also published in: J.W. Shavlik and T.G. Dietterich (Eds.) *Readings in Machine Learning, 184-191*. Morgan Kaufmann, 1990.

[15] T.M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[16] J.R. Quinlan and R. Cameron-Jones. Oversearching and layered search in empirical learning. *Proc. 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*, 1019-1024. Morgan Kaufmann, 1995.

[17] R.B. Rao, D. Gordon, and W. Spears. For every generalization action, is there really an equal and opposite reaction? Analysis of the conservation law for generalization performance. *Proc. 12th Int. Conf. on Machine Learning,* 471-479. Morgan Kaufmann.

[18] C. Schaffer. A conservation law for generalization performance. *Proc. 11th Int. Conf. on Machine Learning, 259-265.* Morgan Kaufmann.

[19] S. Singh. Anomaly detection using negative selection based on the r-contiguous matching rule. *Proc. 1st Int. Conf. on Artificial Immune Systems (ICARIS-2002)*, pp. 99-106. University of Kent at Canterbury, UK, Sep. 2002.

[20] G. Stanfill and D. Waltz. Towards memory-based reasoning. *Communications of the ACM, 29(12)*, 1213-1228, Dec. 1986.

[21] A.B. Watkins and L. Boggess. A resource limited artificial immune system classifier. *Proc. Congress on Evolutionary Computation (CEC-2002)*. IEEE Press.

[22] A. Watkins and J. Timmis. Artificial Immune Recognition System (AIRS): revisions and refinements. *Proc. 1st Int. Conf. on Artificial Immune Systems (ICARIS-2002)*, pp. 173-181. University of Kent at Canterbury, UK, Sep. 2002.

[23] I.H. Witten and E. Frank. *Data Mining: practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann, 2000.