Adaptive clusters formation in an Artificial Immune System

Sławomir T. Wierzchoń[1,2)], Urszula Kużelewska[2)]

[1)] Institute of Computer Science, Polish Academy of Sciences 01-237
Warszawa, ul. Ordona 21

[2)] Department of Computer Science, Białystok Technical University
15-351 Białystok, ul. Wiejska 45[a]
e-mail: stw@ipipan.waw.pl

A new version of an artificial immune system designed for automated cluster formation in training data is presented. The algorithm fully exploits self-organizing properties of the vertebrate immune system and produces stable immune network.

Keywords: immune network, clonal selection, somatic mutation, cluster formation

1. Introduction

From a computer science perspective the immune system is a complex, self organizing and highly distributed system which has no centralized control and which uses learning and memory when solving particular tasks. The learning process does not require negative examples and the acquired knowledge is represented in explicit form. These features attract computer scientists offering new paradigm for information processing. Particularly, Jerne's idea of the immune network, [7], has been used to develop new tools for data analysis. The first paper devoted to this subject was that of Hunt and Cooke, [6], where the idea of the immune network with cells represented by the binary strings has been applied to the recognizing promoters in DNA sequences. Next this idea was refined by Timmis, [10], who proposed an interesting algorithm for unsupervised learning, data analysis and data visualization. Here instead of binary representation of antigens and antibodies (see Section 2 for explanation of these terms) real vectors were used. De Castro and von Zuben, [3] developed another algorithm for data analysis and data reduction that refers to the meta-dynamics typical for the so-called second generation immune networks, [11]. Watkins, [12], applied these ideas to the supervised learning. Although interesting, these solutions are not free from disadvantages. The approach presented in [3] is an exciting data reduction technique offering a compact representation of the training data. To extract clusters in the resulting set, the authors use a mixture of standard clustering techniques however. On the other hand the system presented by Timmis in [10] is unstable in the sense that it is hard to keep network size within reasonable boundaries through all the iterations.

In this paper an improved immune algorithm is presented which admits two useful properties: (i) the network size does not exceed the size of the training set in all the iterations, (ii) stable clusters are formed. Before presenting new algorithm (Section 3) we briefly review the main

immune system properties (Section 2). Next numerical experiments are reported (Section 4) and main features of the algorithm are stressed (section 5).

2. Basic facts from immunology

The aim of this section is brief presentation of the main mechanisms used by the immune system. A reader interested in a deeper review of natural immune system from a mathematical perspective is referred to [9] while computer-oriented treatment of the problem can be found in [2] or [13].

The basic building blocks of the immune system are white blood cells, or lymphocytes. There are two major classes of lymphocytes: B-cells, produced in the *bone marrow* in the course of so-called clonal selection (described later), and T-cells, processed in the *thymus*. In the artificial immune systems designed for data analysis problems, the B-cell is the fundamental object for machine learning. Hence, in the sequel we will focus on the properties and mechanisms that govern B-cell populations.

B-cells synthesize and carry on their surfaces molecules called *antibodies* which act like detectors. The specialized portion of the antibody molecule used for identifying other molecules is called *paratope*. Being a 3-D structure with uneven surface the paratope have a unique shape which is referred to as the *specificity*. The regions on any molecule that the paratopes can attach to are called *epitopes*. If the two colliding molecules have complementary specificities, they bind to each other and the strength of the bond (called *affinity*) depends on the degree of complementarity. A molecule bound by an antibody is referred to as the *antigen*[1]. A crucial role of the immune system is the binding of antibodies with antigens which serves to tag them for destruction by other cells. This process is termed *antigen recognition*. To treat formally the recognition problem, Perelson [8] introduced the notion of the *shape space*. Namely, if there are m features influencing the interaction between the molecules (i.e. the length, height, width, charge distribution, etc.) and D_i, $i = 1, \ldots, m$ is the domain of i-th feature then a point in m-dimensional space $S = D_1 \times \ldots \times D_m$ is the generalized shape of a molecule. Typically S is a subset of m-dimensional Hamming space, or m-dimensional Euclidean space.

When a B-cell recognizes an antigen, it may be stimulated to clone (i.e. producing identical copies of itself) as well as to secrete free antibodies. This process of amplifying only those cells that produce a useful antibody type is called *clonal selection*. The number of clones produced by a lymphocyte is proportional to its stimulation level. Clones are not perfect, but they are subjected to *somatic mutation* (characterized by high mutation rate) that result with children having slightly different antibodies than the parent. These new B-cells can also bind to antigens and if they have a high affinity to the antigens they in turn will be activated and cloned. The rate of cloning a cell is proportional to its "fitness" to the problem: fittest cells replicate the most. The somatic mutation guarantees sufficient variation of the set of clones, while selection is provided by competition for pathogens. The whole process of (in fact Darwinian) selection and differentiation of B-cell receptors leading to the evolution of B-cell populations better adapted to recognize specific epitopes is said to be *affinity maturation*.

Besides somatic mutation the immune system uses a number of other mechanism to maintain sufficient diversity and plasticity. Particularly about five percent of the B-cells are replaced

[1] Antigen means a molecule that causes *anti*bodies *gen*eration.

every day by new lymphocytes generated in the bone marrow. This process is termed *apoptosis*.

The immune system possesses two types of response: primary and secondary. The *primary response* occurs when the immune system encounters the antigen for the first time and reacts against it. To learn the structure of the antigen epitopes, affinity maturation is used. The primary response can take some time (usually about 3 weeks) to destroy the antigen. If the body is reinfected with a previously encountered antigen, it will have an adapted subpopulation of B-cells to provide a very specific and rapid *secondary response*. Usually it is very fast and efficient. From a computer science perspective the primary response corresponds to the identification of clusters in the training data, while the secondary response – to the pattern recognition problem, i.e. the assignment of a new data into one of existing clusters. Interestingly, the secondary response is not only triggered by the re-introduction of the same antigens, but also by infection with new antigens that are similar to previously seen antigens. That is why we say that the immune memory is associative.

The final immune system principle that plays a useful role in designing artificial immune system is that of *immune network* theory formulated by Jerne, [7], and further developed by Perelson, [8]. According to this theory (called also *Jerne's hypothesis*) the immune response is based not only on the interaction of B-cells and antigens but also on the interactions of B-cells with other B-cells. These cells provide both a stimulation and suppression effect on one another and it is partially through this interaction that the memory is retained in the immune system.

The immune system is in permanent flux. The whole network is subjected structural perturbations through appearance and disappearance of some cell species. The introduction of new species is caused by somatic mutation, apoptosis, or combinatorial diversity (e.g. genetic operations). A crucial issue is the fact that the network as such, and not the environment, exerts the greatest pressure in the selection of the new species to be integrated in the network. Thus, the immune network is self-organizing, since it determines the survival of newly created clones, and it determines its own size. This is referred to as the *meta-dynamics* of the system, [11].

The model of immune memory proposed by Jerne resembles the models of hypercycles or autocatalytic sets considered in the context of prebiotic chemical evolution – cf. [1] or [4]. It seems that careful examination of these models may be of value in constructing effective data analysis algorithms.

3. The algorithm

The algorithm used in the experiments and depicted on Figure 1 resembles Timmis' algorithm presented in [10]. Some of its steps are implemented differently, however.

In the algorithm we use a stylised version of the immune system. In the natural system each B-cell has about 10^5 antibodies attached to its surface. Since all these antibodies have identical paratopes, in artificial immune systems we identify each cell with its antibody which is represented by an m-dimensional real vector. To compare degrees of affinity between different antibodies and antigens it is assumed that the vectors representing these molecules are normalized, i.e. both Ag (the set of antigens) and Ab (the set of antibodies) are actually subset of m-dimensional unit cube $[0, 1]^m$. The elements of Ag are normalized during the first

presentations of the antigens to the system (step 1 of the algorithm). While in [10] each element of the training set was assigned randomly (with equal probability) to the set *Ag* or *Ab*, in our approach the set *Ag* is identical with the training set, and the set of antibodies is initiated randomly.

```
1. Present antigens set, Ag. Let n = |Ag|
2. Generate randomly antibodies set, Ab, of size n.
3. Find an initial value of the NAT scalar.
4. loop
     4.1. Find stimulation level
     4.2. Purge worst cells
     4.3. Rebuild network
     4.4. Recalculate NAT scalar
     4.5. Clone and mutate cells
5. until (termination-condition)
```

Figure 1. The algorithm for an immune network generation

The affinity between two cells ab_j, $ab_k \in Ab$, $j \neq k$, is expressed as the Euclidean distance $d_{jk} = d(ab_j, ab_k)$. If the distance is not greater than a pre-specified threshold value, called the network affinity threshold or *NAT*, then a link is created between the two cells. The *NAT* is a very important parameter as it determine the granularity of the network and its overall connectivity. In [10] an initial value of the *NAT* was defined as the average distance between all the cells in *Ag*. However such a procedure overestimates the actual threshold value what results in collapsing the network into a single cluster as the number of iterations increases. In our approach the *NAT* scalar is computed in step 3 as the average distance between $n \cdot \kappa$ lowest distances between the antigens. Here κ is a parameter and in all our experiments $\kappa = 1$. (Note that when $\kappa = n$, we obtain the original definition used in [10]). In step 4.4 the threshold value is computed as the average length of the $n \cdot \kappa$ shortest links.

Stimulation level, $sl(ab_j)$, is the second very important factor. In [10] it was computed in accordance with Jerne's hypothesis as the sum of three subfactors: (i) affinity among a given antibody and all the antigens, (ii) the degree of stimulation of the antibody by other antibodies from the networks, and (iii) the degree of suppression of the antibody by other antibodies. Since antigens and antibodies are points in the $[0, 1]^m$, the computation of the subfactors (ii) and (iii) seems to be slightly artificial since we do not distinguish between paratopes and epitopes. Thus the stimulation level is defined in terms of its affinity to the set of antigens only. That is given $d_{jk} = d(ab_j, ag_k)$ the set of distances between j-th antibody and k-th antigen we find the minimal distance $d_{j\cdot} = \min\{ d_{jk} | k = 1, \ldots, n\}$. Now $sl(ab_j) = 1 - d_{j\cdot}$ if $d_{j\cdot} \leq NAT$ and $sl(ab_j) = 0$ otherwise. Such a strategy reasonably differentiates the antibody cells contrary to the Timmis' approach.

An antibody with stimulation level $sl(ab_j)$ produces $\lfloor c \cdot sl(ab_j) \rfloor$ clones, where c is a constant (maximal number of clones). Each clone $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ is subjected mutation according to the equation

$$y_i = y_i + r \cdot \Delta,\ i = 1, \ldots, m$$

where r is a random number from the unit interval and $\Delta = 1 - y_i$ or $\Delta = -y_i$ where the decision which value to choose is made randomly.

4. Experiments

Three training data sets were analysed – see figures (2a), (3a) and (4a). The corresponding immune networks are presented at figures (2b), (3b) and (4b).

Interestingly, to classify correctly presented antigens, only few iterations suffice. Figures (2c), (3c) and (4c) show evolution of the fraction of correctly recognized cells as well as the evolution of the network sizes in subsequent iterations. Typically, after 10 iterations all the antigens are properly classified and the network size tends to the size of the antigens set. This is in contrast with Timmis' approach, where the network size increases with iterations.

Another interested feature of the model is the adaptive tuning of the NAT scalar – see figures (2d), (3d) and (4d). The initial NAT value is usually small

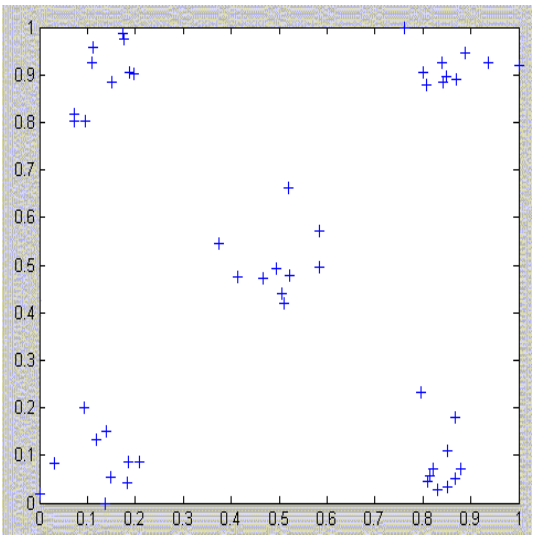


Fig.2 a) Antigens set

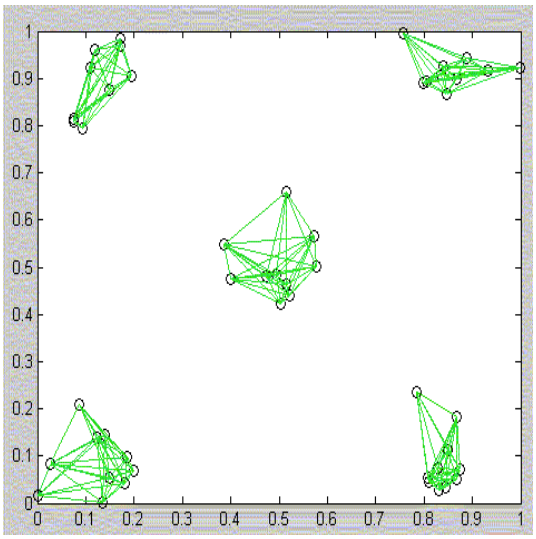


Fig. 2b) Final immune network

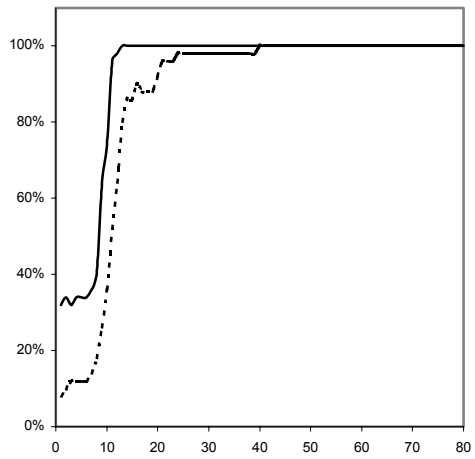


Fig. 2c) Number of correctly recognized cells and the network size

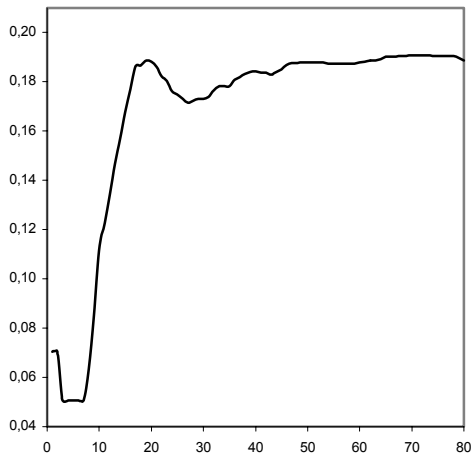
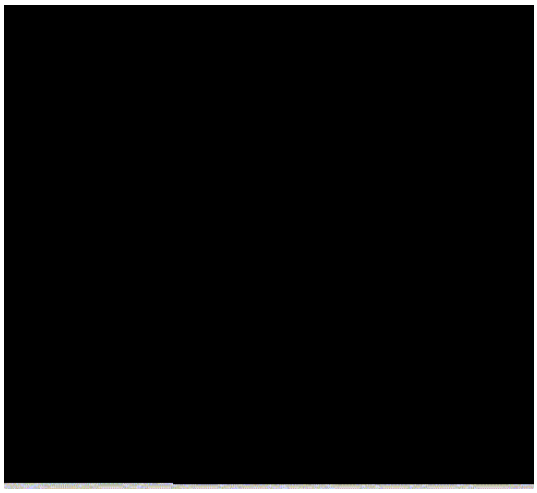
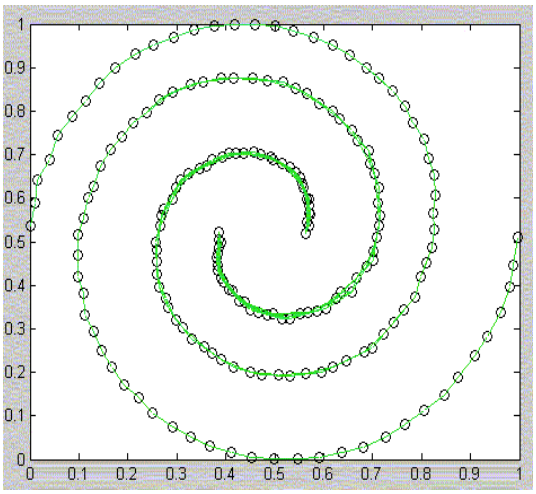


Fig. 2d) Evolution of the NAT scalar



3a) Antigens set



3b) Final immune network

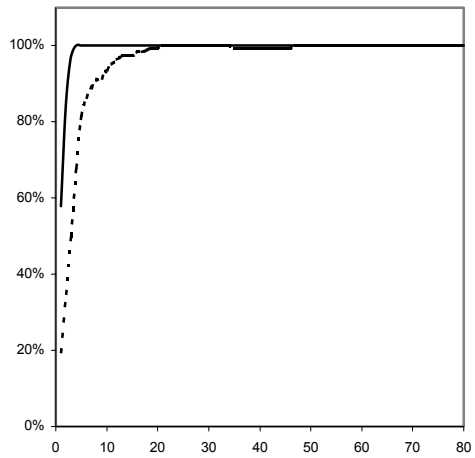


Fig. 3c) Number of correctly recognized cells and the network size

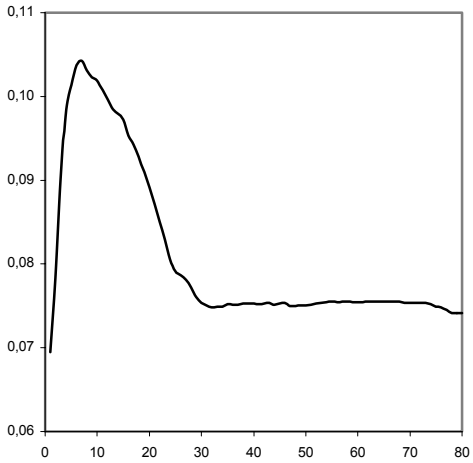


Fig. 3d) Evolution of the NAT scalar

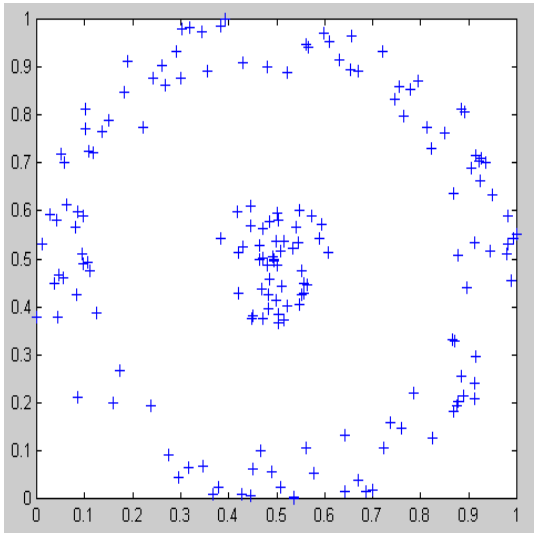


Fig. 4c) Antigens set

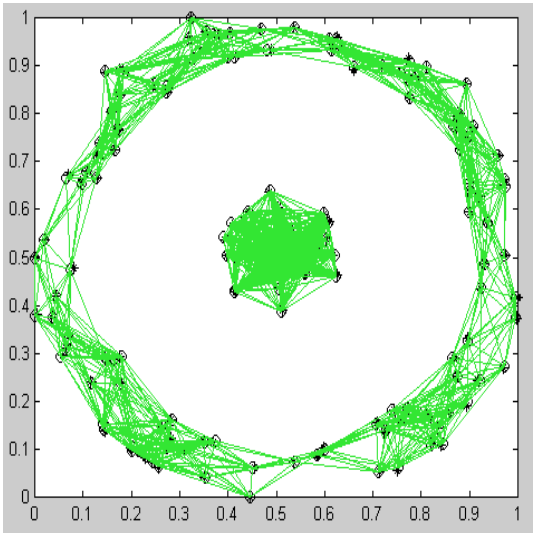
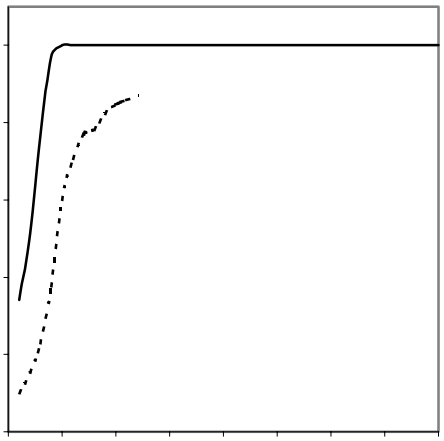


Fig. 4b) Final immune network



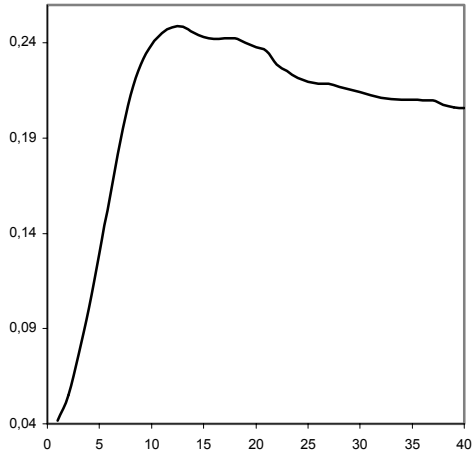


Fig. 4c) Number of correctly recognized cells and network size

Fig. 4d) Evolution of the NAT scalar

[6] Hunt, J.E., Cooke, D.E. Learning using an artificial immune system. *J. of network and Computer Applications*, **19**(1996)189-212
[7] Jerne, N.K. Towards a network theory of the immune system. *Ann. Immunol* (*Inst. Pasteur*) **125C**(1974)373-389
[8] Perelson, A.S. Immune network theory. *Immunological Reviews*, **110**(1989)5-33
[9] Perelson, A.S., Weisbuch, G. Immunology for physicists. *Reviews of Modern Physics*, **69**(1977)1219-1265
[10] Timmis, J.I. Artificial immune systems: A novel data analysis technique inspired by the immune network theory. Ph. D. Thesis. Department of Computer Science, University of Wales, Aberystwyth, September 2000
[11] Varela, F.J., Coutinho, A. second generation immune networks. *Immunology Today*, **12**(1991)159-166
[12] Watkins, A.B. AIRS: A resource limited artificial immune classifier. M. Sc. Thesis. Department of Computer Science, Mississippi State University, December 2001
[13] Wierzchoń, S.T. *Artificial Immune Systems. Theory and Applications* (in Polish). Akademicka Oficyna Wydawnicza EXIT, Warszawa 2001. ISBN 83-87674-30-3, 282+vii pp.