Hacia un Modelo de Red Inmunológica Artificial Basado en Kernels

Towards a Kernel-Based Model for Artificial Immune Networks

Juan C. Galeano, Ing.[1], Fabio A. González, PhD.[2]
[1]Intelligent Systems Research Lab, National University of Colombia - Branch Bogota
[2]Bioingenium Research Lab, National University of Colombia - Branch Bogota
{jcgaleanoh, fagonzalezo}@unal.edu.co

Recibido para revisión 28 de Noviembre de 2007, aceptado 14 de Febrero de 2008, versión final 28 de Febrero de 2008

***Resumen*—En este artículo se presenta una adaptación de un modelo de red inmunológica artificial a la estrategia de los métodos de *kernel*. Esta adaptación proporciona al área de los sistemas inmunológicos artificiales, por primera vez, algunas de las ventajas de los métodos de *kernel*, tales como la habilidad para ser aplicados sobre datos no vectoriales y la transformación a espacios de alta dimensionalidad por medio del llamado *kernel trick*. Algunos experimentos preliminares fueron llevados a cabo con el fin de obtener algunas luces sobre el comportamiento del modelo propuesto.**

Palabras Clave-**sistemas inteligentes, agrupamiento basado en similitud, redes inmunológicas artificiales, métodos de kernel.**

Abstract-**This paper presents an adaptation to the strategy of kernel methods of a well known artificial immune network. This adaptation brings to artificial immune systems, for the first time, some of the advantages of kernel methods, such as the ability to deal with non-vector data and the mapping to high-dimensional spaces through the kernel trick. Preliminary experiments were carried out in order to get some insights of the behavior of the proposed model.**

Keywords-**intelligent systems, similarity-based clustering, artificial immune networks, kernel methods.**

I. INTRODUCTION

Machine learning looks for finding and recognizing patterns in data coming from a source of interest [2]. The patterns extracted from such data may be used for obtaining knowledge of the data source or to make predictions [2].

Machine learning techniques can be classified into, at least, two approaches: the feature-based approach and the similarity-based approach [13]. The former considers a process to represent the objects as feature vectors as a previous step to learning [2]; the latter, uses only a (dis)similarity measure between objects as the input for the learning process [13]. There are some domains where the objects have a complex structure, which makes difficult to perform the feature extraction process [13]. Examples of this kind of domains are images, sequences, graphs, and text documents. In those cases, the application of the similarity-based approach appears to be more suitable.

Kernel methods is a machine learning strategy that makes algorithm design independent from the representation of the data [25]. Such independence requires that algorithms receive as input a pair-wise measure instead of feature vectors. From this point of view we can consider the strategy of kernel methods as a similarity-based approach.

Bio-inspired computing (also known as natural computing) is the field of machine learning (and computer science) that develops problem-solving techniques by mimicking a natural system. Some examples of this kind of techniques are Genetic Algorithms that take inspiration from species evolution [9], Artificial Neural Networks that take inspiration from brain function [4], and Artificial Immune Systems (AIS) that take inspiration from the immune system behavior [6]. Most of those algorithms follow the feature-based approach since the first step in setting them up is a representation process where domain objects are usually coded as attribute vectors, with each component representing some important feature of the object.

There is not much research on the development of bio-inspired algorithms that follow the similarity-based approach, except for the kernel versions of the self-organizing map (kernel SOM) [16], [21], [12] and the kernel particle swarm optimization (kernel PSO) [1]. In this paper, the problem of expressing an artificial immune network (AIN) model as a kernel method is tackled. Specifically, a known AIN model, aiNet [7], is modified in such a way that it does not receive as input the vector representation of antigens, but the inter-antigen distances.

This paper is organized as follows: Section II presents a review of the artificial immune network model for clustering and the strategy of kernel methods for pattern analysis. Section III presents an AIN model for clustering based on the kernel methods strategy. Section IV presents the results of some preliminary experiments, and finally, Section V concludes the paper and proposes some ideas for future work.

II. BACKGROUND

A. Artificial Immune Networks

The natural immune system (NIS) is the main defense of the body against pathogen organisms, called antigens [22]. The NIS exhibits some interesting properties from a computational point of view, such as pattern recognition, autonomy, distributivity, learning and memory [6]. Thus the field of artificial immune systems (AIS) works on the definition of models inspired by principles and theories of the NIS in order to be applied to problem solving [6].

The immune learning is the ability of the NIS for adapting itself to the changes in the antigenic environment [6]. When an antigen enters the body for the first time, the NIS uses adaptation mechanisms to create cells (B-cells) and molecules (antibodies) able to deal with such antigen. Additionally, the NIS uses a memory mechanism for keeping an internal image of the antigen, so that it can deal with it faster and more effectively if the same antigen (or a similar one) invades the body again [22].

From the point of view of the immune network theory, immune learning and memory can be explained as follows: the production of a given antibody (elicited by an external antigen) stimulates/ suppresses the production of other antibodies that stimulate/ suppress the production of other antibodies and so on [22]. The main hypothesis of this theory states that immune memory is maintained by B-cells interacting with each other even in the absence of foreign antigens [22]. These interactions between cells/molecules are via shape matching (recognition). The way an antigen (or an antibody) can stimulate/suppress the production of antibodies is based on the strength of the recognition, which is called affinity [22].

An artificial immune network (AIN) is an AIS that uses concepts from the immune network theory [6], mainly the interactions between B cells (stimulation and suppression) and the cell cloning and mutation processes [8]. In the last years different AIN models have been proposed to solve problems such as pattern recognition in DNA sequences [10], data analysis [20], [15], [27], [26], [7], web mining [18], [19], multimodal function optimization [5], robot control [11], [17], autonomous navigation [14], [17] and e-mail classification [24].

Most of AIN models are based in the shape-space concept introduced by Perelson [8]. This concept represents antibodies and antigens as points in a n-dimensional space, where each dimension is related to a feature involved in the recognition process [22]. An antibody recognizes those antigens in its scope, which is defined by a hypersphere with certain recognition radius and center in the antibody [22]. From this point of view, AIN can be labeled as a feature-based approach.

Most of AIN models assume a feature based representation of the domain objects to be used in the learning process. This makes difficult to apply such models in domains with complex structure. Then, a representation independent AIN model would allow to extend the application domain of AIN. The kernel methods approach gives a suitable strategy to achieve such model.

B. Kernel Methods

Kernel methods are described as a unified approach based on statistical methods for pattern analysis [25]. The approach assumes that transforming the input data to a proper feature space could make the pattern identification process easier [25]. Such transformation corresponds to the representation process mentioned above. However, the transformation is defined implicitly by a kernel function, which is related to the specific data type and the kind of patterns that is expected to find in the data set [25].

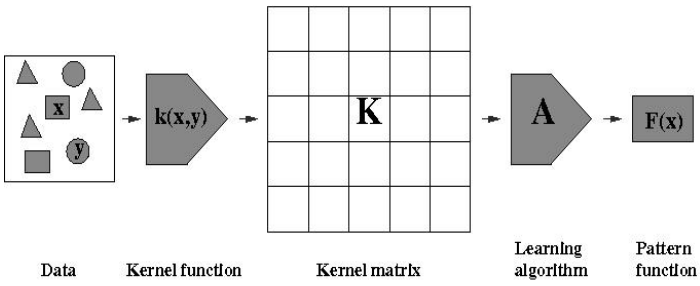


Fig. 1. A schematic representation of the kernel methods approach

The kernel methods strategy can be represented as in Fig. 1. Data are mapped to a feature space through a kernel function κ, which computes dot products between objects in the new space. The kernel matrix K contains the evaluation of the kernel function for each pair of objects and acts as input of a learning algorithm A that is designed to find patterns in the feature space based on such matrix [25]. One interesting and useful property of the kernel function is that it can compute the dot product from the original objects. This property is called the kernel trick [23]. Then, the learning algorithm is designed so that it does not need the coordinates of the objects in the new space, but only the kernel matrix [25].

Kernel methods can be applied to any data type, *i.e.*, input data do not need to be represented as feature vectors [25]. It is because constraints to consider a function as a kernel function are stated in terms of the kernel matrix. Therefore, any pair-wise measure (*e.g.* similarity) able to produce a pair-wise matrix that

satisfies those constraints can be considered as a kernel function [25]. Hence, kernel methods can be labeled as a similarity-based approach.

Kernel methods have the advantage of separating the learning algorithm from the data representation [25]. This allows research on pattern analysis to be defined in two directions: one for defining kernel functions for specific data types, and other for building learning algorithms to find specific pattern types.

The definition of a kernel method implies the expression of computations in terms of dot products [25]. This can be achieved directly as the case of support vector machines or by modifying a known model as the case of the kernel k-means, which is the kernel version of the k-means algorithm [25].

C. Kernel Methods on Bio-inspired Algorithms

There is not much research on the development of bio-inspired algorithms that follow the kernel method strategy. Here, we review the kernel versions of the self-organizing map (SOM) and the kernel version of the particle swarm optimization algorithm (PSO).

1) *KM-Kernel SOM*: In [16] MacDonald and Fyfe present a kernel version of the Kohonen's self-organizing map (SOM). The authors review the kernel *k*-means (KM), which is the base for this kernelized SOM (KSOM). In that model, each mean m_μ is a linear combination of input data in the feature space, which is expressed as

$$m_\mu = \sum_i \gamma_{\mu i} \phi(x_i)$$

where $\{x_i\}_{i\in[1,N]}$ is the set of input samples, ϕ is the mapping from the input space to the feature space, induced by a kernel function, and $\gamma_{\mu i}$ is the weight of the *i*-th simple for the mean m_μ. Coefficients $\gamma_{\mu i}$ are in [0, 1] and the sum over all the coefficients representing a mean equals the number of points assigned to the cluster that mean represents. Thus, in KM-KSOM, neuron weights are linear combinations of input data in the feature space, and the weight updating mechanism is the same as in the original SOM, even using the same neighborhood function. As reviewed in [12], in this model neuron weights are defined in the feature space but not in the input space.
2) *GD-Kernel SOM*: This kernel version of SOM, reviewed in [12], is presented in [21] by Pan *et al.* and in [3] by Andras. The model is based on gradient descent (GD) and the problem is to minimize the mean square error of an input and the corresponding prototype in the mapped space.
3) *EF-Kernel SOM*: In [12] Lau *et al.* present a review of the KM-KSOM the GD-KSOM. They show that SOM can be performed completely in feature spaces and that both versions of KSOM are equivalent to a unifying KSOM based on energy functions, all of this keeping in mind classification tasks. Authors derive a KSOM based on a energy function (EF) to be minimized. It is followed from a definition of the original SOM as minimizer of such function. They then show the updating rules of KM-SOM and GD-SOM are particular cases of the updating rule in the EF-SOM, that can be used depending on weights are or are not defined in the feature space.
4) *Kernel PSO*: In [1] Abraham *et al.* present a kernel version of the Particle Swarm Optimization (PSO) algorithm to perform automatic clustering, *i.e.*, without prior knowledge about the right number of clusters. The modified version is achieved by defining a clustering index, called CS measure, in terms of inner products and solving the clustering problem as an optimization problem where CS measure should be minimized, which implies an optimal partition has been reached. In the process, each particle represents a partition of the input data set and therefore, the solution is the globally best particle at the end of the process.

III. A PROPOSAL FOR A KERNEL ARTIFICIAL IMMUNE NETWORK (KAIN)

Fig. 2 shows a general training algorithm for AIN, which is taken from [8]. The algorithm receives as input a set $A \subset \boldsymbol{A}$ of antigens, which are going to be presented to the network, where $\boldsymbol{A}$ is the set of all possible antigens. The algorithm returns an artificial immune network composed of a set $B \subset \boldsymbol{B}$ of B-cells and connections between them, where $\boldsymbol{B}$ is the set of all possible B-cells. The set B is the set of B-cells of an AIN at a given time.

The first step, initialization, is to create an initial set of B-cells. After this, an iterative process is performed starting by presenting the set of antigens to the network, which means that for each antigen and each B-cell the stimulation is calculated ($f^A_{stimulation}: B \times A \to \Re$). Such stimulation is based on an affinity measure between antigens and B-cells ($f_{affinity}: B \cup A \times B \cup A \to \Re$) that measures the similarity/ complementarity between elements in the shape-space.

In the next step B-cells are allowed to interact with each other, this is done by calculating the stimulation and suppression effects between them ($f^B_{stimulation}: B \times B \to \Re$ and $f^B_{\sup pression}: B \times B \to \Re$). Similar to antigen/B-cell stimulation, B-cell/B-cell stimulation (and suppression) could be calculated as a function of B-cell/B-cell affinity. Total stimulation of B-cells is then calculated by summing up the effects caused by antigen and network interactions ($F: B \to \Re$). Based on total stimulation, some B-cells are selected and some copies ($f_{cloning}: B \to \mathrm{N}$) of each selected B-cell are created and mutated ($mutate: B \to B$). Some models interpret this rate as the probability of a B-cell to be selected for suffering mutation; other models interpret it as the

GAIN(A: antigen set)
1: *initialization*
1.1: assign B an initial set of B-cells
1.2: initialize network structure L
2: repeat until a stop criterion is met
2.1: *antigen presentation:*
2.1.1: calculate $f_{affinity}(a,b)$ for all $a \in A, b \in B$ ▷ *Antigen/B-cell affinity*
2.1.2: calculate $f^{\mathbb{A}}_{stimulation}(b,a)$ for all $a \in A$ and $b \in B$ ▷ *Antigen/B-cell stimulation*
2.2: *B-cell interaction:*
2.2.1: calculate $f^{\mathbb{B}}_{stimulation}(b,b')$ and $f^{\mathbb{B}}_{suppression}(b,b')$ for all $b,b' \in B$
▷ *B-cell/B-cell stimulation/suppression*
2.3: *affinity maturation:*
2.3.1: calculate $F(b)$ as

$$F(b) = \sum_{a \in A} f^{\mathbb{A}}_{stimulation}(b,a) + \sum_{b' \in B, b' \neq b} f^{\mathbb{B}}_{stimulation}(b,b') \quad \triangleright \textit{ Total stimulation}$$
$$+ \sum_{b' \in B, b' \neq b} f^{\mathbb{B}}_{suppression}(b,b'), \text{ for all } b \in B$$

2.3.2: repeat $f_{cloning}(b)$ times ▷ *cloning and mutation*
2.3.2.1: $b' = mutate(b)$ ▷ *create a clone and mutate it*
2.3.3: calculate stimulation of all new B-cells
2.4: *meta-dynamics:*
2.4.1: update network structure L ▷ *deletion/creation of B-cells and links*
3: return (B, L) ▷ *Return immune network*

Fig. 2. A General Artificial Immune Network algorithm

proportion of the B-cell fields that will be changed.

In the meta-dynamics step, some useless B-cells are removed from the network, new B-cells are created randomly and incorporated into the network, and links among all B-cells are reorganized. Finally, when the stopping criterion is met, the current network is returned.

A A Modified Version of the GAIN Algorithm

To modify the GAIN algorithm so that it follows the strategy of kernel methods, it should be noticed that a crucial step is the mutation operator, since it is usually implemented as a few variation in the coordinates of a feature vector. For this purpose we propose to define a new representation for B-cells.

1) *Representation and Mutation of B-Cells*: Here, we follow a similar approach to that of Kernel-SOM. Generally, in most AIN models, antigens correspond to input samples and antibodies may be assimilated to cluster centroids (neural weights). Therefore, in KAIN B-cells (antibodies) are represented as linear combinations of antigens in the feature space. In a more formal way, a B-cell is represented as

$$b_i = \sum_j \gamma_{ij} \phi(a_j)$$

where $b_i \in B$ is a B-cell, $a_j \in A_i$ is an antigen, ϕ is a mapping from the input space to a feature space and γ_{ij} is a weight. We can constrain the linear combination to be convex in order to make the B-cell be surrounded by the antigens it recognizes. In general, a_j may be any antigen presented to the network, *i.e.* $A_i = A$, however we can restrict A_i to include only those antigens which have interacted with (stimulated) b_i. Notice that those A_i are not necessarily disjoint.

From the biological point of view, this representation makes sense because when an antigen enters the body, the immune system creates some B-cells in order to neutralize such antigen. The B-cells more likely to survive are those with high affinity with the stimulating antigen, in that way, those B-cells are internal images of such antigen.

As a B-cell is stimulated by those antigens close enough in the shape-space, *i.e.*, the affinity between the B-cell and each of those stimulating antigens is greater than a certain threshold, the shape of a B-cell is defined for the set of antigens it recognizes, as more than one B-cell can recognize the same antigen, the sets A_i are not necessarily disjoint.

Notice that given a set of antigens $A = \{a_j\}_{j \in [1, n]}$, a B-cell b_i is totally determined by the coefficients γ_{ij}. This

means that a B-cell b_i can be represented as the vector of weights $(\gamma_{i1} \ldots, \gamma_{im})$. Based on this representation it is easy to adapt a genetic algorithm mutation scheme for real-valued chromosomes. For instance, a Gaussian mutation scheme as described in [9]

$$\begin{aligned} mutate(b_i) &= mutate(\gamma_{i1},\ldots,\gamma_{im}) \\ &= (\gamma_{i1},\ldots,\gamma_{im}) + (\Delta_1,\ldots,\Delta_m) \end{aligned}$$

where each $\Delta_i \sim N(0, \sigma)$ and σ is a parameter that controls the expected size of Δ_i. Notice that a normalization step is required after adding the vector Δ in order to keep the convexity property. The implementation of this operator is straightforward, however the crucial step is the implementation of mutation in such a representation.

2) B-cell ↔ antigen and *B-cell ↔ B-cell affinity*: As it was mentioned in Section II, affinity is a central concept in AIN models. Since B-cells are represented as a linear combination of antigens, the B-cell ↔ antigen and the B-cell ↔ B-cell affinity may be defined in terms of antigen ↔ antigen affinity. Notice that in the natural immune system there is not such a concept as antigen ↔ antigen affinity; this is also the case in most AIN. However, most of the AIN represent B-cells and antigens in the same way, thus the B-cell ↔ antigen affinity measure may be naturally extended to an antigen ↔ antigen affinity measure.

For this paper, we use the antigen ↔ antigen affinity given by

$$f_{affinity}(\phi(a_i), \phi(a_j)) = \exp\left(-\frac{D_{ij}^A}{\sigma^2}\right)$$

where $D_{ij}^A = \|\phi(a_i) - \phi(a_j)\|^2$

Given that $\|x\|^2 = \langle x, x \rangle$, the expression above turns into

$$\begin{aligned} D_{ij}^A &= \langle \phi(a_i) - \phi(a_j), \phi(a_i) - \phi(a_j) \rangle \\ &= \langle \phi(a_i), \phi(a_i) \rangle + \langle \phi(a_j), \phi(a_j) \rangle - 2\langle \phi(a_i), \phi(a_j) \rangle \end{aligned}$$

which can be expressed using the kernel function κ as

$$D_{ij}^A = \kappa(a_i, a_i) + \kappa(a_j, a_j) - 2\kappa(a_i, a_j)$$

In a similar way, the B-cell ↔ antigen affinity will given by

$$f_{affinity}(b_i, \phi(a)) = \exp\left(-\frac{D_{ia}^{BA}}{\sigma^2}\right)$$

where

$$\begin{aligned} D_{ia}^{BA} &= \|b_i - \phi(a)\|^2 \\ &= \sum_j \gamma_{ij} \sum_k \gamma_{ik} \kappa(a_j, a_k) + \kappa(a, a) + \\ &\quad - 2\sum_j \gamma_{ij} \kappa(a_j, a) \end{aligned}$$

and finally, the B-cell B-cell affinity can be expressed as

$$f_{affinity}(b_i, b_j) = \exp\left(-\frac{D_{ij}^B}{\sigma^2}\right)$$

where

$$\begin{aligned} D_{ij}^B &= \|b_i - b_j\|^2 \\ &= \langle b_i - b_j, b_i - b_j \rangle \\ &= \langle b_i, b_i \rangle + \langle b_j, b_j \rangle - 2\langle b_i, b_j \rangle \end{aligned}$$

and using the fact that

$$\begin{aligned} \langle b_i, b_j \rangle &= \left\langle \sum_k \gamma_{ik} \phi(a_k), \sum_l \gamma_{jl} \phi(a_l) \right\rangle \\ &= \sum_k \gamma_{ik} \sum_l \gamma_{jl} \langle \phi(a_k), \phi(a_l) \rangle \\ &= \sum_k \gamma_{ik} \sum_l \gamma_{jl} \kappa(a_k, a_l) \end{aligned}$$

the expression above turns into

$$\begin{aligned} D_{ij}^B &= \sum_k \gamma_{ik} \sum_l \gamma_{il} \kappa(a_k, a_l) + \\ &\quad \sum_k \gamma_{jk} \sum_l \gamma_{jl} \kappa(a_k, a_l) + \\ &\quad - 2\sum_k \gamma_{ik} \sum_l \gamma_{jl} \kappa(a_k, a_l) + \end{aligned}$$

3) New B-cell generation: As the representation of B-cells in feature space requires a set of antigens for each B-cell, the initialization step can be performed by randomly selecting a set $A_0 \subset A$ where each antigen represents a B-cell, *i.e.*, if the *k*-th antigen is selected, the corresponding B-cell will be $b = \sum_j \gamma_j \phi(a_j)$ where $\gamma_j = 0$ for all $j \neq k$ and *1* for $j = k$.

IV. PRELIMINARY EXPERIMENTS

Some simple experiments were carried out in order to test the suitability of the proposed model for detecting clusters in data. In this work results will be validated by visual evaluation of B-cells locations. For this purpose a synthetic *2*-dimensional data set consisting of *5* clusters was used. This data set is a version of the one used in [7], it consists of *50* points, each cluster containing *10* points. Fig. 3 shows a plot of this data set. The implementation we used is an adaptation of the aiNet algorithm [7] considering the definitions presented in Section III. One major modification was introduced in the affinity measure, while aiNet algorithm uses *1 - D* as the affinity measure, where *D* represents the Euclidean distance between patterns, the proposed algorithm uses $\exp\left(-\frac{D^2}{\sigma^2}\right)$ as the affinity measure.

A Experimental Setup

In this setup *2* experiments were carried out varying the input data: the first one using clusters *2* and *3* (points in the upper left hand and lower right hand), and the second one using the whole data set. Table I shows an explanation of each parameter along with the values used in this setup.

For these experiments, the identity kernel $\kappa(x, y) = \langle x, y \rangle$ was used, which means that no transformation of the input space is made. This has the purpose of the visual validation of the algorithm's performance.

B Experimental Results

Fig. 4 and Fig. 5 show the output of the algorithm for the experiments performed. Antigens are represented as circles and B-cells are represented as crosses. Notice that in both experiments, B-cells are located in the antigen regions, and regions free from antigens remain empty. This suggests that the algorithm is able detect clusters. Another property that can be seen is that the number of B-cells is low compared to the number of antigens. Notice that this algorithm does not perform data compression, as the original aiNet does, because each antigen presented to the network is included in a linear combination that defines a B-cell.

Taking into account these results, we can say that it performs well in spaces where clusters appear to be well separated. As the model has been presented as a kernel method, we could take advantages from the kernel trick in order to transform spaces of input data with different cluster structures into spaces where they appear to be well separated.

V. CONCLUSION

In this paper a kernel-based model for artificial immune networks was introduced. A key concept for this model is the representation of antibodies as linear combinations of antigens, which allows the definition of a mutation mechanism without assuming a vector representation of antibodies. The model assumes an affinity measure between antigens, which is an idea present in the current models and supported by the shape-space concept, but not explicitly mentioned.

Experimental results on synthetic data gave a good insight of the ability of the proposed model for detecting clusters. Some issues were mentioned regarding the modified version of the aiNet model, which suggest some tasks for future work, namely

- to test the use of the kernel trick with different kinds of cluster structures,
- to include the idea of "antibody saturation", which means that a B-cell can be defined by a limited number of antigens, and
- to apply the model to a non-vector data set.

ACKNOWLEDGMENT

Authors thank to members of the Intelligent Research Lab. and Bioingenium Research Lab. for their support in this project.

REFERENCES

[1] Ajith Abraham, Swagatam Das, and Amit Konar. Kernel based automatic clustering using modified particle swarm optimization algorithm. In Dirk Thierens and et al., editors, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pages 2-9, University College London, Gower Street, London, UK, July 2007. ACM.

[2] E Alpaydin. Introduction to Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2004.

[3] P. Andras. Kernel-Kohonen networks. Int. J. Neural Syst., 12:117-135, 2002.

[4] Christopher M. Bishop. Neural Networks for Pattern Recognition. Oxford University Press, 1995.

[5] L. N. De Castro and J. Timmis. An artificial immune network for multimodal optimisation. In In Congress on Evolutionary Computation. Part of the 2002 IEEE World Congress on Computational Intelligence, pages 699 - 704, Honolulu, Hawaii, USA, May 2002. IEEE.

[6] L. N. De Castro and J. Timmis. Artificial Immune Systems: A New Computacional Intelligence Approach. Springer-Verlag, 2002.

[7] L. N. de Castro and F. J. V. Zuben. aiNet: An Artificial Immune Network for Data Analysis. In H. A. Abbas R. A. S. and Charles S. Newton, editors, Data Mining: A Heuristic Approach, chapter XII, pages 231 - 259. Idea Group Publishing, USA, 2001.

[8] J. C. Galeano, A. Veloza-Suan, and F. A. González. A comparative analysis of artificial immune network models. In H. Beyer, editor, Proceedings of GECCO 2005, pages 361 - 368. ACM Press, 2005.

[9] David E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, 1989.

[10] J. E. Hunt and D. E. Cooke. Learning using an artificial immune system. Journal of Network and Computer Applications, 19:189 - 212, 1996.

[11] A. Ishiguro and Y. Uchikawa. A gait acquisition of sixlegged robot using immune networks. In Proceedings of the International Conference on Intelligent Robotics and Systems (IROS?94), volume 2, pages 1034 - 1041, Munich, Germany, 1994.

[12] King Wai Lau, Hujun Yin, and Simon Hubbard. Kernel selforganising maps for classification. Neurocomputing, 69:2033-2040, 2005.

[13] R. S. Ledley and C. Y. Suen. Pattern recognition, the journal of the pattern recognition society, October 2006.

[14] G.-C. Luh and W.-W. Liu. Reactive immune network based mobile robot navigation. In G. NicosiaV. CutelloP. J. Bentley. Timmis, editor, Proceeding of the Third Conference ICARIS, pages 119 - 132. Springer, 2004.

[15] Neal. M. Meta-stable memory in an artificial immune network. In P. Bentley J. Timmis and E. Hart, editors, Proceedings of the Second International Conference ICARIS, pages 168 - 180, Edinburg, UK, September 2003. Springer.

[16] D. MacDonald and C. Fyfe. The kernel self organising map. In Proc. 4th int. conf. on knowledge-based intelligence engineering systems and applied technologies, pages 317-320, 2000.

[17] R. Michelan and F. J. Von Zuben. Decentralized control system for autonomous navigation based on an evolved artificial immune network. In Proceedings of the IEEE Congress on Evolutionary Computation, volume 2, Honolulu, HI, May 2002. IEEE.

[18] O. Nasraoui, C. Cardona, C. Rojas, and F. González. Tecnostreams: Tracking evolving clusters in noisy data streams with a scalable immune system learning model. In Third IEEE International

Conference on Data Mining,Melbourne, FL, November 2003. IEEE.

[19] O. Nasraoui, F. González, and D. Dasgupta. The fuzzy artificial immune system: Motivations, basic concepts and application to clustering and web profiling. In IEEE International Conference on Fuzzy Systems, pages 711 - 716, Hawaii, HI, May 2002. IEEE.

[20] M. Neal. An artificial immune system for continuous analysis of time-varying data. In J. Timmis and P. J. Bentley, editors, Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS), volume 1, pages 76 - 85, University of Kent at Canterbury, September 2002. University of Kent at Canterbury Printing Unit.

[21] Z. S. Pan, S. C. Chen, and D. Q. Zhang. A kernel-base SOM classifier in input space. Acta Electron. Sin., 32:227-231, 2004. in Chinese.

[22] A. S. Perelson and G. Weisbuch. Immunology for physicists. Reviews of Modern Physics, 69(4):1219 - 1267, October 1997.

[23] Bernhard Schlkopf and Alexander J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond. The MIT Press, December 2001.

[24] A. Secker, A.and Freitas and J. Timmis. Aisec: An artificial immune system for e-mail classification. In H. Abbass T. Kay-Chen R. McKay D. Essam R. Sarker, R. Reynolds and T. Gedeon, editors, Proceedings of the Congress on Evolutionary Computation, page 131 ? 139, Canberra. Australia, December 2003. IEEE.

[25] J. Shawe-Taylor and N. Cristianini. Kernel Methods for Pattern Analysis. Cambridge University Press., 2004.

[26] J. Timmis, Neal M., and J. Hunt. An artificial immune system for data analysis. BioSystems, 55:143 - 150, 2000.

[27] J. Timmis and M. Neal. A resource limited artificial immune system for data analysis. Knowledge-Based Systems, 14:121 - 130, 2001.

Juan Carlos Galeano. Earned his bachelor's degree in computer science and engineering from the National University of Colombia at Bogota, and he is currently a master student in computer science and engineering at the same university.

Fabio A. González. Earned his bachelor's degree in computer science and engineering, and his master's degree in mathematics from the National University of Colombia at Bogota. He also earned a master's degree and a PhD degree in computer science from the University of Memphis, USA. He is currently an Associate Professor of the Systems and Computer Engineering Department at the Nacional University of Colombia.

Universidad Nacional de Colombia Sede Medellín
Facultad de Minas

Escuela de Ingeniería de Sistemas

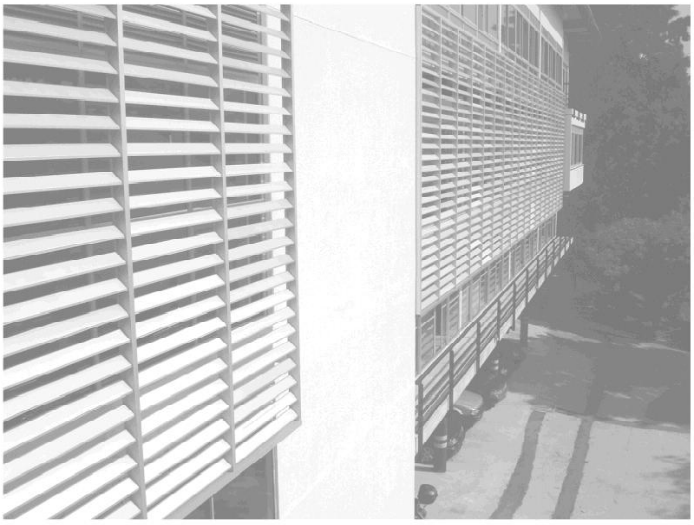
Grupos de Investigación

Grupo de Investigación en Sistemas e Informática

Categoría A de Excelencia Colciencias
2004 - 2006 y 2000.

GIDIA: Grupo de Investigación y Desarrollo en Inteligencia Artificial

Categoría A de Excelencia Colciencias
2006 – 2009.



Grupo de Ingeniería de Software

Categoría C Colciencias 2006.

Grupo de Finanzas Computacionales

Categoría C Colciencias 2006.

Centro de Excelencia en Complejidad

Colciencias 2006

Escuela de Ingeniería de Sistemas
Dirección Postal:
Carrera 80 No. 65 - 223 Bloque M8A
Facultad de Minas. Medellín - Colombia
Tel: (574) 4255350 Fax: (574) 4255365
Email: esistema@unalmed.edu.co
http://pisis.unalmed.edu.co/