Institute of Research and Publication Indonesia
IJIRSE: Indonesian Journal of Informatic Research and Software Engineering
Journal Homepage: https://journal.irpi.or.id/index.php/ijirse
Vol. 5. No.2. September 2025, pp: 107 - 117
E-ISSN: 2775-5754 | P-ISSN: 2797-2712

Design of Artificial Immune System - Models and Algorithms

Teguh Sujana[1*], Chairun Nas[2]

[1]Departement of Information System, Universitas Riau, Indonesia

E-Mail: [1]teguh.sujana@lecturer.unri.ac.id, [2]chairun.nas@lecturer.unri.ac.id

Makalah: Received July 20th 2025; Revised July 31th 2025; Accepted August 05 th 2025
Corresponding Author: Teguh Sujana

Abstract

Artificial Immune Systems (AIS) belong to a group of computational intelligence methods inspired by the working mechanisms of biological immune systems to solve various computational problems. Artificial Neural Networks (ANNs) themselves are often used in various fields such as anomaly detection, pattern recognition, cyber and network security, task scheduling, process optimization, and data analysis, with the application of various ANN algorithms. In the AIS approach, there are four basic algorithms that serve as the main foundation, namely the Negative Selection Algorithm (NSA), Artificial Immune Networks (aiNet), Clonal Selection Algorithm (CLONALG), and Dendritic Cell Algorithm (DCA). The problem that occurs at this time is that there is still a lack of papers that discuss the main basic algorithms in AIS, resulting in difficulties in developing new models of basic algorithms. Apart from that, many other aspects of the natural immune system have not been touched due to not yet understanding the basic algorithm of AIS. This paper aims to explain the main models and algorithms in AIS above so that in future research, new algorithms can be developed based on the basic algorithm as a reference. The results of this paper are a review of the main basic models and algorithms in AIS.

Keyword: Artificial Immune System, Negative Selection, Artificial Immune Network, Clonal Selection, Dendritic Cell.

1. INTRODUCTION

Along with the passage of time, knowledge such as in the field of computation is continuously developed to address the challenges faced by society. These challenges encompass areas such as health computation, economics and trade, decision-making, and others. Researchers continue to advance their expertise and attempt to adopt various real-life examples currently available, much like incorporating diverse performance processes from the human body. An illustrative instance of knowledge adopted from the human body's performance system, currently popular, includes human intelligence, neural networks, and the latest addition being the adoption of insights from the human immune system. Through the assimilation of such knowledge, the field of computation gives rise to disciplines like Artificial Intelligence (AI), Artificial Immune Systems (AIS), and Artificial Neural Networks (ANN). With these developed disciplines, they are integrated into computation, enabling computers to address issues akin to human bodily functions. This paper delves into the adoption of knowledge, specifically focusing on Artificial Immune Systems.

Artificial Immune System (AIS) is an intelligent computational model that imitates the functioning of the human immune system, characterized by autonomy, learning, memory, adaptation, resilience, and scalability[1]. AIS utilizes signaling, learning, and memory to accomplish tasks of classification and pattern recognition, retaining patterns learned previously[2]. AIS finds extensive application in computer security, anomaly detection, Web Mining, the Internet of Things (IoT), Numerical Function Optimization, and Combinatorial Optimization[3]. Generally, there are four primary fundamental algorithms within AIS. These algorithms include Clonal Selection Algorithm (CLONALG), Dendritic Cell Algorithm (DCA)Artificial Immune Network Algorithm (aiNet),and Negative Selection Algorithm (NSA)[4]. It is these foundational algorithms that researchers continuously develop further to address existing problems and identify shortcomings within the basic algorithms.

In principle, AIS adopts the working system of the immune system, where the immune system will detect any foreign objects that enter the body. The immune system will react to these foreign objects to be recognized, and automatically, cells within the immune system will form to destroy these foreign objects and store them in memory. If at some point these foreign objects reappear, the immune system can quickly recognize and destroy

them. From this working system, it is applied to the algorithms within AIS, which can detect viruses, identify computer security vulnerabilities, perform classifications, and more. This paper will discuss the concepts and models of each of the aforementioned basic algorithms so that in future research, basic algorithms within AIS can be developed to address the limitations of the foundational algorithms.

2. MATERIALS AND METHOD

2.1 Immune System

The human immune system safeguards the body against various types of pathogens, such as harmful bacteria and viruses, that can infiltrate the body[1]. The immune system comprises several bioactive molecules, cytokines, and proteins, collectively forming a diverse biochemical network that can defend against pathogens[5]. To combat viruses and harmful bacteria, the immune system must be capable of recognizing and identifying pathogens, each of which possesses molecules known as antigens[6]. Antigens are unique structures on pathogens that allow the immune system to recognize different types. When pathogens enter the body, the immune system responds through two main mechanisms[7]. An overview of the immune system response presented in Figure 1.

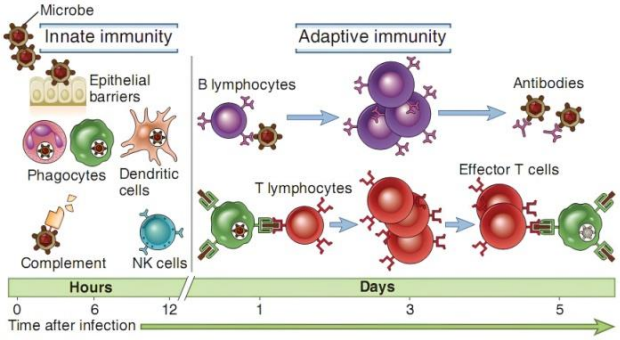

Figure 1. The immune system responds[8].

1. Innate Immune Respond, this acts as the first line of defense in the early stages of infection, until the pathogen is effectively eliminated. However, under certain conditions, these defenses may fail due to the high intensity or number of invading pathogens. In such situations, lymphocytes and adaptive immune mechanisms are activated for the specific recognition and elimination of pathogens[8].
2. Adaptive Immune Respond, represents an evolution within the immunoglobulin family (antibodies) and cells such as B lymphocytes (B cells) and T lymphocytes (T cells)[8]. T cells control the adaptive immune response and destroy pathogens and infected cells. Meanwhile, B cells produce antibodies against specific antigens. Antibodies are proteins that bind to pathogens. Through these proteins, immune cells can be signaled to destroy pathogens[3].

2.2 Artificial Immune System

The Artificial Immune System (AIS) is an adaptive system that mimics the function of the human immune system as a model principle in problem-solving for fields such as reinforcement learning, artificial neural networks, classification learning systems, computer security, Web Mining, numerical function optimization, and genetic algorithms[9]. Essentially, the immune system has several properties, namely[10].

1. Detection: This occurs within the immune system when infectious fragments chemically bind to sensory receptors on the surface of lymphocyte cells.
2. Diversity: This is related to non-self bodies of organisms in the immune system, leading the immune system to possess a variety of sensory receptors where certain lymphocytes will react to foreign organisms.
3. Learning: The ability to swiftly detect and eliminate foreign organisms from the body.
4. Tolerance: This mechanism refers to particles that mark them as part of the body and are stored in the chromosomes.
5. Uniqueness: Each individual processes their immune system using unique vulnerabilities and capabilities.
6. Recognition of Foreign Organisms: The immune system detects and eliminates harmful molecules that do not originate from the body.

AIS is categorized into two domains, namely optimization and classification, with two types of algorithm categories: population-based category, where there are Clonal Model and Negative Selection algorithms, and network-based category, where there are Continuous Model and Discrete Model algorithms [3]. Clonal Model algorithms are commonly used for optimization, while Negative Selection is applied to classification and

clustering. Network-based algorithms are commonly utilized for classification. The categories of Artificial Immune Systems are presented in Figure 2.

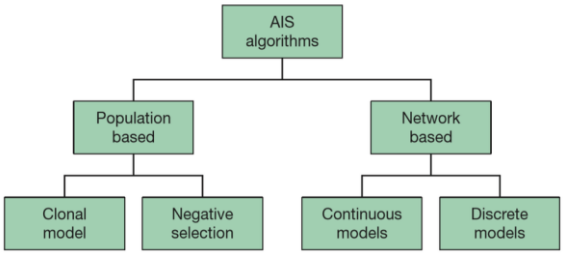

Figure 2. Artificial Immune System Categories[3].

This section will explain the concept of basic algorithms in AIS, namely, Negative Selection Algorithm (NSA), Clonal Selection Algorithm (CLONALG), Artificial Immune Network (aiNet), dan Dendritic Cell Algorithm (DCA).

2.3 Negative Selection Algorithm

The Negative Selection Algorithm (NSA) aims to generate immune detection, such as computer security, network security, and anomaly detection, based on the 'Random-Discard' model by classifying data, referred to as antigens [11]. Essentially, NSA is a large-scale maturation detector (antibody) generated randomly, and then the portion covering the self area is discarded (apoptosis) in self-reactive T cells. If the T-cell set detects self-area cells, these cells are eliminated, and immune function is carried out in the process of T-cell maturation. Meanwhile, detectors that cannot detect self-antigens will be retained[4]. There is a basic definition in processing the NSA algorithm, namely:

1. Antigen Device, The antigen set can be defined as A_g = {x1, x2, x3, …., xn}, in which $x_i \in$ [0,1], n denotes the total number of sample points, x_i denotes the normal value of the sample point i, A_g refers to the normalized set of values of all sample points.
2. Self-Antigen and Nonself-Antigen. Self-Antigen is defined as self $\in A_g$, representing positive samples, and Nonself-Antigen, defined as nonself = A_g, representing negative samples. The area covered by the Self-Antigen in the range of value space is referred to as the Self-Region, and the area that is not covered is referred to as the Nonself-Region.
3. Affinity. Euclidean distance $dist(x_i,x_j) = \sqrt{\sum_{d-1}^{D}(x_i^d - x_j^d)^2}$ between two points refers to the measure of affinity that connects them, where x_i and x_j denotes the i-th sample points and j-th sample points, d denotes the feature dimensions of the sample points, D denotes the total number of feature dimensions covered by the sample points and is the feature of the d-th dimension of the sample point i-th.
4. Detector. The detector is denoted by $d_e(z_i,r_i)$, where r_i denotes the center of a randomly generated detector candidate, r_i signifies the distance from this center to the nearest self-cell. The circle defined by z_i and r_i corresponds to the maturity level of the detector.

An overview of the process of detector generation to produce data detection in the NSA is presented in Figure 3.

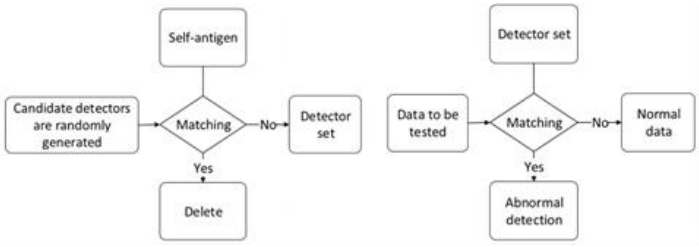

Figure 3. (a) Detector Generation (b) Data Detection[4].

NSA has limitations as it does not take into account the unbalanced distribution of antigens in the sample space, leading to detectors overlapping and causing substantial redundancy. To address this limitation, in 2003, the Real-Valued Negative Selection Algorithm (RNSA) was developed, which distributes detectors in the non-self area based on heuristics to optimally maximize coverage area[11]. The advantages of RNSA include increased expressive capabilities, the potential to gain high-level knowledge from the resulting detectors, and increased scalability under certain conditions. The algorithmic form of RNSA is as follows [12].

Input : The Self Training set (*Train*), The Radius of Detector (r_d), The number of needed detector *maxNum*.
Output : The Detector Set *D*.
Step 1 : Initialize the self training set *Train*.
Step 2 : Randomly generate a candidate detector d_{new}. Calculate the euclidean distance between d_{new} and all the selves in *Train*. If $dis(d_{new}, ag) < rd + rs$ for at least one self antigen ag, execute Step 2; if not, execute Step 3.
Step 3 : Add d_{new} into the detector set *D*.
Step 4 : If the size of *D* satisfies $N_d > maxNum$, return *D*, and the process ends; if not, jump to step 2.

RNSA is continuously developed to enhance the distribution of its detectors for greater optimization. The development algorithms stemming from RNSA include the Grid-Based Real-Valued Negative Selection Algorithm, the Voronoi Diagram Negative Selection Algorithm, and the Antigen Density Clustering-Based Negative Selection Algorithm[13].

2.4 Artificial Immune Network Algorithm

The Artificial Immune Network (aiNet) algorithm is designed to reduce and classify separate data by forming a network of interconnected antibodies based on affinity levels. From this network, a subset of antibodies with the highest affinity for the antigen is selected and cloned in proportion to its value, forming an inversion layer according to its affinity. A certain percentage of the cloning results are designated as memory antibodies. If two memory antibodies have affinities exceeding the threshold, one of them will be removed from the network. Conversely, memory antibodies with affinities to the antigen below the threshold will be eliminated. [14]. An illustration of the aiNet algorithm can be seen in Figure 4.

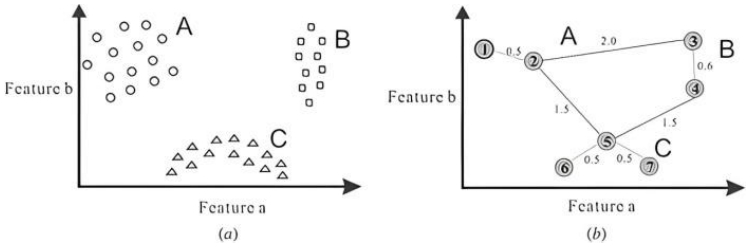

Figure 4. The illustration of the aiNet algorithm[14].

Figure 4(a) shows a dataset consisting of three dense clusters (A, B, C). Figure 4(b) shows each cluster forming a cell network, with lines representing connections to link separate clusters and classify different groups within the network. The numbers on the memory cells indicate labels, while the numbers on the lines represent connection strengths. Generally, the number of memory cells is greater than the number of clusters, but still far fewer than the number of samples[15]. The process flow of the aiNet algorithm can be explained as follows [14]:

Input : Antibody Network (*Ab*), Suppression Threshold (δ_s), Natural Death Thershold (δ_d), rate of affinity (ζ), Number of best-matching cells taken for each Ag_i (*n*), Clone Number Multiper (*N*).
Output : The Worst Individual (r%) and Number of Iterations.
Step 1 : Create a random initial population from *Ab* and initialize the parameters.
Step 2 : Clone selection: for each antibody, determine its affinity with the presented antigen.
Step 3 : Select (n%) of the highest affinity tissue cells.
Step 4 : Reproduction (*n*) of clones of selected cells. The number of offspring of each cell, N_c, is proportional to its affinity is $f_{i,j} = 1 / D_{i,j}$. Where $D_{i,j}$is the inequality calculated according to the euclidean distance, the higher the affinity, the larger the clone size. The total clone size N_c of the resulting for each cell in *Ab* is obtained using the formula:

$$Nc = \sum_{i=1}^{n} round\left(N - D_{i,j} * N\right) \quad (1)$$

Step 5 : Mutation of each antibody is inversely proportional to affinity, resulting in the mutation set $C_k^*$with the formula:

$$C_k^* = C_k + \alpha_k \left(Ag_j - C_k\right);\ \alpha \propto 1 / f_{i,j};$$
$$k = \{1, \ldots, N_c\} \quad (2)$$

Step 6 : Calculate the affinity of the antibody enhanced with the antigen.
Step 7 : Reselect ζ% of the best antibody (Highest Affinity), place it into the clone memory set.
Step 8 : Remove affinity antibodies (*Ab* - *Ag*) with an antigen yield lower than the δ_d threshold (pruning threshold).
Step 9 : Calculate the affinity of tissue cells (*Ab* - *Ag*).
Step 10 : Combine the remaining antibodies from the clone memory with all the tissue antibodies.

Step 11 : Determine the entire intercellular affinity network and eliminate antibodies whose affinity for one another is lower than the suppression threshold (δ_s).
Step 12 : Replace r% of the worst individuals with randomly generated novels.
Step 13 : Repeat step 2 until step 4 until the predefined number of iterations is reached.

The development of the aiNet algorithm is the Optimization Artificial Immune Network Algorithm (Opt-aiNet), which aims for data processing, optimization learning, and fault diagnosis, enabling noise handling, unsupervised learning, and self-organization. The Opt-aiNet algorithm, in the process of tracking local and global optima, must select all cells for cloning in each iteration to maintain local optima, thus requiring exploration of the entire space[16]. An overview of the antibody population transformation process in the Opt-aiNet algorithm can be seen in Figure 5.

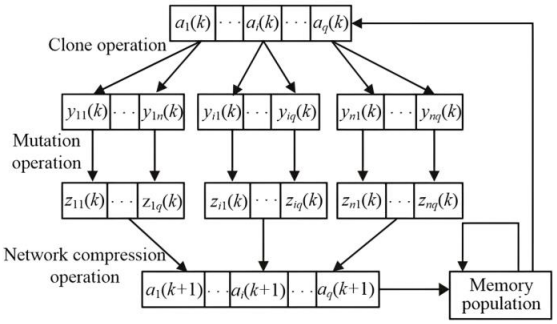

Figure 5. Transformation of the antibody population for the Opt-aiNet algorithm[17].

Several algorithms were developed from this Opt-aiNet Algorithm, such as the Artificial Immune Network algorithm for Combinatorial Optimization (copt-aiNet). This algorithm is associated with the suppression mechanism to find similarities between cells by calculating the minimum number of swap operations required to convert the resulting solution[18]. Further more, there is the Artificial Immune Network algorithm for Dynamic Optimization (dopt-aiNet) which enhances the robustness of opt-aiNet in quickly handling dynamic problems. This algorithm is designed to improve the mutation operator and suppression mechanism in the opt-aiNet algorithm[19].

2.5 Clonal Selection Algorithm

The Clonal Selection Algorithm (CLONALG) was developed with the intention of addressing multimodal and combinatorial optimization problems using the principle of clone selection[20]. CLONALG shares similarities with the immune system in the human body in terms of specificity, proliferation (cloning), and variation. In the immune system, after being stimulated by antigens, the immune system will produce B cells that can generate antibodies. Antibodies effectively eliminate antigens through proliferation (cloning) and variation. Antibodies are specific, where the effectiveness of different antibodies depends on the specific antigen they are exposed to. B cells are stimulated to become memory cells, so when the same antigen is detected in the future, antibodies are rapidly produced to counteract the antigen[21]. The basic principle of applying the clone selection algorithm can be seen in Figure 6.

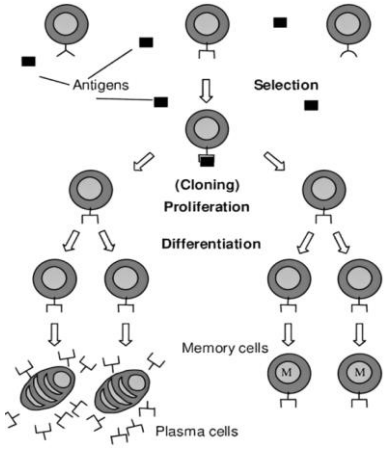

Figure 6. The principle of clonal selection[22].

The process flow of the CLONALG algorithm can be explained as follows [22].

Input : The best n antibodies to be selected (*n*), The size of the population (*N*), the affinity of an antibody (*F*), The number of copies (*nc*), The number of randomly generated antibodies (*d*).
Output : Number of Iterations.
Step 1 : Initialize *N* antibody set size, number of iterations, number of clones, and relevant parameters. Randomly select an antigen from the antigen set and generate a candidate antibody set consisting of a memory set and a residue set.
Step 2 : Calculate the *F* difference between the respective antibody and antigen in the concentration of the candidate antibody, then select the antibody with the highest number.
Step 3 : Cloning antibody *n* and the number of antibody clones that have a positive correlation with their affinity for an antigen.
Step 4 : Mutates the antibodies produced after cloning to create new individuals. The likelihood that the mutation is influenced by antibody affinity, the higher the level, the lower the chance of an antibody mutation.
Step 5 : Count the antibodies after the mutation, then select the highest antibody to compare with the antibodies in the memory set. The highest antibody compared will be entered into the memory set.
Step 6 : Evaluate the mutated antibody. Randomly select mutated antibodies to replace with old n antibodies. Then the d antibodies that have been randomly selected are added to the next generation population.
Step 7 : Repeat step 2 for the next iteration. If the number of iterations satisfies the termination conditions, the process stops.

The CLONALG algorithm is continuously developed to address its deficiencies. For instance, the Clonal Selection Classification Algorithm (CSCA) introduces self-adjustment capabilities, insensitivity to parameters, and competitiveness as a classification system for binary pattern recognition datasets. Furthermore, the Adaptive Clonal Selection (ACS) algorithm represents an evolution of parameter-free CLONALG. This algorithm is applied in the domain of static function optimization. Additionally, there exists the Lamarckian Clonal Selection Algorithm (LCSA), which substitutes the mutation process in CLONALG with local search techniques[23].

2.6 Dendritic Cell Algorithm

The Dendritic Cell Algorithm (DCA) is an algorithm developed by mimicking the danger theory of cells within the body, which involves signals of damage originating from endogenous sources and tissue cells themselves. This ensures that the immune system does not respond to its components but rather to potential threats[24]. In danger theory, biologically, cells that are in a threatened state will release endogenous signals called danger signals, establish danger zones in the vicinity. Antigens within this zone are captured by antigen presenting cells, such as macrophages, and then transported to lymph nodes to be presented to lymphocytes. B cells that produce antibodies will search for matching antigens in the danger zone to trigger stimulation and clonal expansion, while mismatched antigens will not induce stimulation[25]. As for the description of the danger theory model is presented in Figure 7.

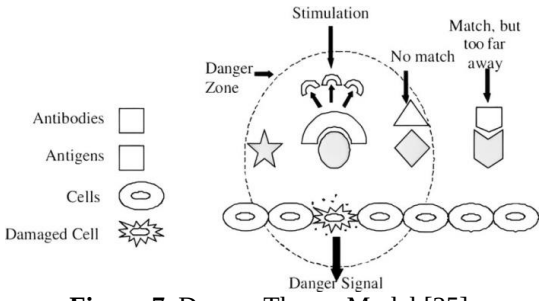

Figure 7. Danger Theory Model [25].

The Dendritic Cell Algorithm (DCA) performs intrusion detection based on a population of dendritic cells, where some dendritic cells are randomly selected to present antigens (sample data) with an unstable number of dendritic cells. Subsequently, the antigens are classified as either normal states or anomalous states. DCA is widely implemented for anomaly detection, such as network anomalies, virus anomalies, and more. Data in DCA can be categorized into three signals[26]:

1. Pathogen Associated Molecular Pattern (PAMP). It is a signal indicating the presence of anomalies associated with the given data instance (antigen).
2. Danger Signal (DS). It is a signal indicating an abnormal state where the given data instance is below PAMP.
3. Safe Signal (SS). It is a signal indicating a normal state with the given data instance above PAMP.

An overview of the Dendritic Cell algorithm process is shown in Figure 8.

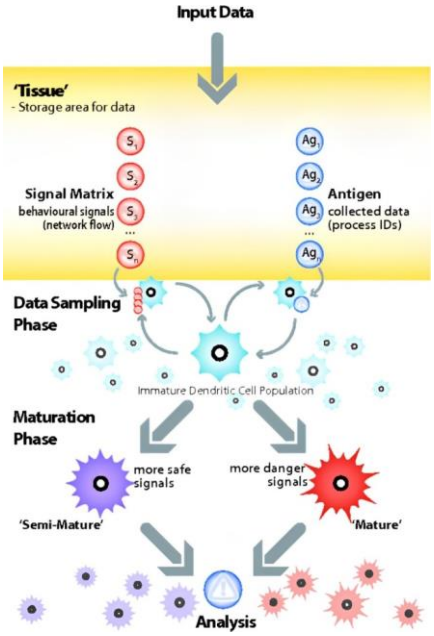

Figure 8. Dendritic Cell Algorithm Process [26].

The process flow of the DCA algorithm can be explained as follows [26]:

Input : Data Set Training, Pathogen Associated Molecular Pattern ($PAMP_i$), Danger signal(DS_i), Safe Signals (SS_i).

Output : Classification of normal and anomaly data.

Step 1 : **Preparation and Initialization**, DCA will select the most important features from the input training dataset and assign signal categories to be PAMP, SS, and DS.

Step 2 : **Detection**, DCA creates a signal database by combining input signals with antigens using the get-antigen and get-signal functions. From the input signals, a provisional output is generated, representing the concentration value of the costimulatory molecular signal (CSM), the semi-mature signal value (smDC), and the mature signal value (mDC). Therefore, to compute the provisional output signal, the following equation is employed:

$$C = \frac{\left((W_{PAMP} * \sum_i PAMP_i) + (W_{SS} * \sum_i SS_i) + (W_{DS} * \sum_i DS_i)\right)}{(W_{PAMP} + W_{SS} + W_{DS})} * \frac{1+I}{2}$$

Step 3 : **Context Assesment**, If the value of smDC is greater than mDC, then DC will be in a semi-mature context (context = 0, normal); if it is smaller, then DC will be in a mature context (context = 1, anomaly).

Step 4 : **Clasification**, The calculated values in the cell context are represented by the mature context antigen value (MCAV). MCAV is computed by dividing the number of antigens present in the mature context, referred to as Nb-mature, by the total number of antigen presentations, referred to as Nb-antigen. Subsequently, a comparison is conducted between the MCAV of each antigen and an anomaly threshold. The anomaly threshold can be a parameter set by the user or can be automatically generated from training or testing data.

One of the developments in DCA is the QuickReduct Dendritic Cell Algorithm (QR-DCA), which aims to improve the signal categorization in order to optimize the classification technique of DCA, because it is able to achieve a balance between optimal classification result quality and increased algorithm flexibility regarding execution time.

3. RESULT AND DISCUSSION

This section presents the findings and provides an in-depth discussion of the applied methods and obtained results.

3.1 Negative Selection Algorithm

A case example involves data logs of activity on a server, used by the NSA to detect anomalous activities that might indicate an intrusion. Normal data ('self') are the daily activity logs of the server that have been

categorized as non-suspicious. The NSA algorithm is then used to generate detectors capable of recognizing abnormal activity logs ('non-self').

First, generate detectors by extracting daily log data from the server, resulting in 1000 random detectors. Next, examine each detector against the normal dataset. Remove detectors that match the normal dataset and save the detectors that do not match the normal dataset.

Once the detectors are obtained, the next step is to use the remaining detectors to monitor new activity logs. The result is that the algorithm will mark logs that match the detectors as anomalies. The process overview of the NSA algorithm is shown in Table 1.

Table 1. Process of NSA Anomaly Detection on the Server

Step	Number of Detector	Detector Removed	Remaining Detector
Initial Generation	1000	-	1000
After Dataset Check	1000	950	50
Monitoring New Data	50	-	50

The visualization result of NSA algorithm process regarding anomaly detection on daily server logs is shown in Figure 9.

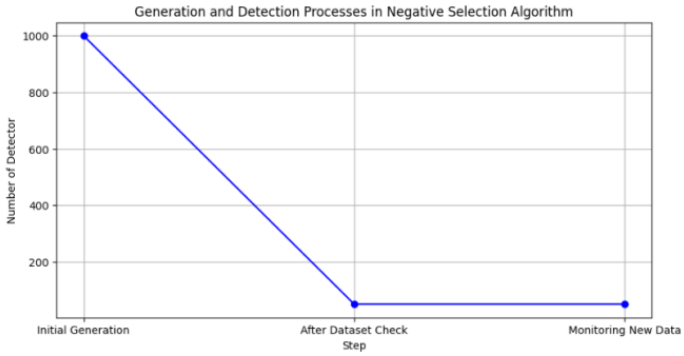

Figure 9. Anomaly detection graph on server daily log data

3.2 Artificial Immune Network Algorithm

In a given scenario, A company has a communication network consisting of 8 nodes and various paths that data packets can take to be sent from the source node to the target node. The AiNet algorithm is then used to find the optimal route that minimizes cost and latency. The data representation of Nodes and paths to be processed using the AiNet is shown in Table 2.

Table 2. Node & Path Representation

Node	Connected Paths	Cost	Latency
1	2, 3	$10	5 ms
2	1, 3, 4	$15	8 ms
3	1, 2, 4, 5	$12	6 ms
4	2, 3, 5, 6	$18	10 ms
5	3, 4, 7	$20	12 ms
6	4, 7, 8	$25	15 ms
7	5, 6, 8	$22	13 ms
8	6, 7	$17	9 ms

The results of optimizing the AINET algorithm for both Node and Path can be visualized in Figure 10.

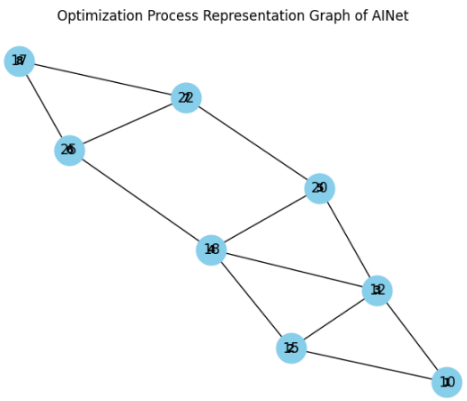

Figure 10. AINET Optimization

The graph illustrates the evolutionary process undertaken by AiNet in finding the optimal route. At each iteration, AiNet refines the previous solution by adapting effective antibodies and eliminating ineffective ones. Ultimately, AiNet generates the best route that meets the criteria of minimal cost and low latency.

Therefore, AiNet can be effectively utilized in routing optimization in communication networks and other optimization problems by drawing inspiration from the mechanisms of the human immune system.

3.3 Clonal Selection Algorithm

A case example where parameter optimization is performed for a machine learning model. Initially, there is a small population of random solutions representing the parameter settings. Each solution is evaluated based on its performance against the objective function, such as model accuracy. The best solutions are then replicated through a clonal process, where individuals with better performance will produce more clones of themselves. This process results in a new, higher-quality population. Below is an example showing the population of solutions and their performance before and after clonal process in Table 3.

Table 3. Solution Population and Performance Before and After the Clonality Process

Solution	Parameter 1	Parameter 2	Initial Performance	Clone
Solution 1	0.5	0.7	0.85	2
Solution 2	0.3	0.9	0.75	1
Solution 3	0.8	0.6	0.92	3

After the clonality process, the new population will consist of better solutions with potentially greater parameter variation. The graph below shows the comparison of solution performance before and after the clonality process as shown in Figure 11.

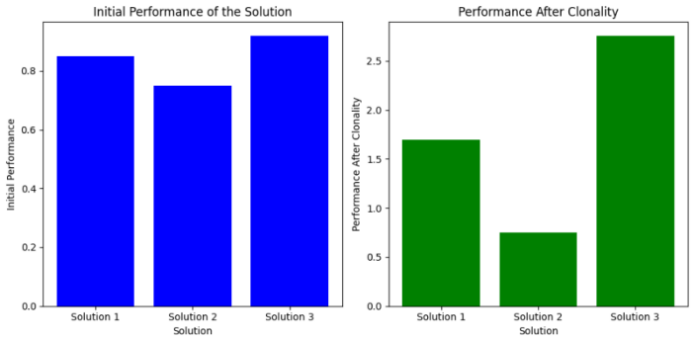

Figure 11. Visualization of performance before and after the clonality process

In the graph, it can be seen that after the clonal process, the performance of the solutions has significantly improved, indicating that the Clonal Selection Algorithm (CSA) has successfully enhanced the quality of the solution population. Thus, the Clonal Selection Algorithm is an effective method for improving and optimizing solutions across various problems, including in the context of machine learning model parameter optimization.

3.4 Dendritic Cell Algorithm

In this case example, DCA (Dendritic Cell Algorithm) is applied to detect anomalies in computer network traffic. Network data is categorized into three types of signals: Danger Signal, Safe Signal, and Contextual Signal.

After implementing DCA on the collected network data, the algorithm classifies each network activity based on the received signals. Activities with more danger signals than safe signals will be marked as anomalies. In this test, the network data consists of 1000 activities, of which 100 are simulated anomaly activities. The result shows that DCA successfully detects 90 out of 100 anomaly activities with a false positive rate of 5%. The form of anomaly detection results from the DCA process above is shown in Table 4.

Table 4. Anomaly Detection Results in Computer Network using DCA

Category	Number of Activities	Anomaly Detection	False Positive
Normal Activities	900	45	45
Anomalous Activities	100	90	0

Here is a graph illustrating the distribution of normal and anomaly activities as well as the detection results by DCA as shown in Figure 12.

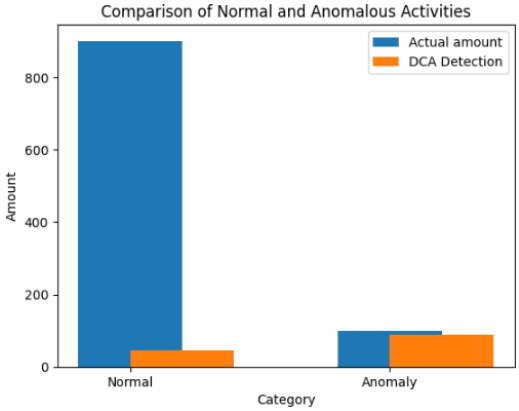

Figure 12. Distribution of Normal and Anomaly Activities

The graphic output illustrates the comparison between the actual amounts of normal and anomalous activities with the detection results by DCA. This graph aids in visualizing the effectiveness of the algorithm in identifying anomalous activities within the computer network.

4. CONCLUSIONS

Based on the design of artificial immune systems, both in terms of models and algorithms, research in this field continues to evolve to produce more optimal models and techniques. The discussion shows that the four basic models and algorithms of artificial immune systems have specific functions. The Negative Selection Algorithm (NSA) is widely used for classification and error detection, particularly in computer security to distinguish between self and non-self entities. NSA forms a set of pattern detectors trained under normal (non-anomalous) conditions to identify new patterns or anomalies. The Clonal Selection Algorithm (CLONALG) is generally applied to optimization and pattern recognition problems, similar to genetic algorithms but without recombination operators, and is effective in scheduling. Artificial Immune Network algorithms are used for clustering, data visualization, and optimization, and can be combined with artificial neural networks, while AiNet is often used to determine optimal travel routes. Dendritic cell algorithms are widely used in intrusion detection in computer security, including port scanning identification, BOTNET activity, and virus detection.

Currently, various models and techniques have been developed using basic artificial immune system algorithms to overcome their limitations. Various case studies have been conducted to demonstrate the application of these algorithms in solving real-world problems and analyzing data. From this perspective, some algorithms have proven to be more suitable for specific application areas. Thus, the development of artificial immune systems is not only based on biological principles and mechanisms but also benefits from integration with other soft computing paradigms such as artificial neural networks, fuzzy logic, genetic algorithms, and other algorithms.

REFERENCES

[1] M. Wang, S. Feng, C. He, Z. Li, and Y. Xue, “An Artificial Immune System Algorithm with Social Learning and Its Application in Industrial PID Controller Design,” *Math. Probl. Eng.*, vol. 2017, no. 1, Jan. 2017, doi: 10.1155/2017/3959474.

[2] S. Golzari, S. Doraisamy, M. N. B. Sulaiman, and N. I. Udzir, “A Review on Concepts, Algorithms and Recognition Based Applications of Artificial Immune System,” 2008, pp. 569–576. doi: 10.1007/978-3-540-89985-3_70.

[3] M. Miralvand, S. Rasoolzadeh, and M. Majidi, “Proposing a features preprocessing method based on artificial immune and minimum classification errors methods,” *J. Appl. Res. Technol.*, vol. 13, no. 4, pp. 477–481, Aug. 2015, doi: 10.1016/j.jart.2015.09.005.

[4] C. J. Delona, P. V Haripriya, and J. S. Anju, “Negative selection algorithm: a survey,” *Int. J. Sci. Eng. Technol. Res.*, vol. 6, no. 4, pp. 711–715, 2017.

[5] P. H. Pandya, M. E. Murray, K. E. Pollok, and J. L. Renbarger, “The Immune System in Cancer Pathogenesis: Potential Therapeutic Approaches,” *J. Immunol. Res.*, vol. 2016, pp. 1–13, 2016, doi: 10.1155/2016/4273943.

[6] A. E. Thompson, “The Immune System,” *JAMA*, vol. 313, no. 16, p. 1686, Apr. 2015, doi: 10.1001/jama.2015.2940.

[7] C. Nas, N. Putra, Y. G. Nengsih, and I. Syafrinal, “Artificial Immune System analysis on Route Construction distribution of gas cylinders using the AINet Algorithm,” *J. Phys. Conf. Ser.*, vol. 1842, no. 1, p. 012001, Mar. 2021, doi: 10.1088/1742-6596/1842/1/012001.

[8] M. G. Netea, A. Schlitzer, K. Placek, L. A. B. Joosten, and J. L. Schultze, "Innate and Adaptive Immune Memory: an Evolutionary Continuum in the Host's Response to Pathogens," *Cell Host Microbe*, vol. 25, no. 1, pp. 13–26, Jan. 2019, doi: 10.1016/j.chom.2018.12.006.

[9] N. Rashid, J. Iqbal, F. Mahmood, A. Abid, U. S. Khan, and M. I. Tiwana, "Artificial Immune System–Negative Selection Classification Algorithm (NSCA) for Four Class Electroencephalogram (EEG) Signals," *Front. Hum. Neurosci.*, vol. 12, Nov. 2018, doi: 10.3389/fnhum.2018.00439.

[10] N. Rai and A. Singh, "Improved Clonal Selection Algorithm (ICLONALG)," *Int. J. Curr. Eng. Technol.*, vol. 5, no. 4, pp. 2459–2464, 2015, [Online]. Available: http://inpressco.com/category/ijcet

[11] F. Zhu, W. Chen, H. Yang, T. Li, T. Yang, and F. Zhang, "A Quick Negative Selection Algorithm for One-Class Classification in Big Data Era," *Math. Probl. Eng.*, vol. 2017, no. 1, Jan. 2017, doi: 10.1155/2017/3956415.

[12] R. Zhang, T. Li, and X. Xiao, "A Real-Valued Negative Selection Algorithm Based on Grid for Anomaly Detection," *Abstr. Appl. Anal.*, vol. 2013, pp. 1–15, 2013, doi: 10.1155/2013/268639.

[13] C. Yang, L. Jia, B.-Q. Chen, and H.-Y. Wen, "Negative Selection Algorithm Based on Antigen Density Clustering," *IEEE Access*, vol. 8, pp. 44967–44975, 2020, doi: 10.1109/ACCESS.2020.2976875.

[14] Y. Li, D. Wang, Y. Yu, and L. Jiao, "An improved artificial immune network algorithm for data clustering based on secondary competition selection," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Jul. 2016, pp. 2744–2751. doi: 10.1109/CEC.2016.7744135.

[15] Y. Zhong, L. Zhang, and W. Gong, "Unsupervised remote sensing image classification using an artificial immune network," *Int. J. Remote Sens.*, vol. 32, no. 19, pp. 5461–5483, Oct. 2011, doi: 10.1080/01431161.2010.502155.

[16] H. N. Agiza, A. E. Hassan, and A. M. Salah, "An Improved Version of opt-aiNet Algorithm (I-opt-aiNet) for Function Optimization," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 11, no. 3, pp. 80–85, 2011.

[17] X. Shi and F. Qian, "A multi-agent immune network algorithm and its application to Murphree efficiency determination for the distillation column," *J. Bionic Eng.*, vol. 8, no. 2, pp. 181–190, Jun. 2011, doi: 10.1016/S1672-6529(11)60024-3.

[18] S. S. F. Souza, R. Romero, and J. F. Franco, "Artificial immune networks Copt-aiNet and Opt-aiNet applied to the reconfiguration problem of radial electrical distribution systems," *Electr. Power Syst. Res.*, vol. 119, pp. 304–312, Feb. 2015, doi: 10.1016/j.epsr.2014.10.012.

[19] D. Q. Vu, V. T. Nguyen, and X. H. Hoang, "An improved artificial immune network for solving construction site layout optimization," in *2016 IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, IEEE, Nov. 2016, pp. 37–42. doi: 10.1109/RIVF.2016.7800266.

[20] C. Yang, B. Chen, L. Jia, and H. Wen, "Improved Clonal Selection Algorithm Based on Biological Forgetting Mechanism," *Complexity*, vol. 2020, pp. 1–10, Apr. 2020, doi: 10.1155/2020/2807056.

[21] W. Luo and X. Lin, "Recent advances in clonal selection algorithms and applications," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, Nov. 2017, pp. 1–8. doi: 10.1109/SSCI.2017.8285340.

[22] X. Wang, A. S. Deshpande, G. B. Dadi, and B. Salman, "Application of Clonal Selection Algorithm in Construction Site Utilization Planning Optimization," *Procedia Eng.*, vol. 145, pp. 267–273, 2016, doi: 10.1016/j.proeng.2016.04.073.

[23] B. S. Rao and K. Vaisakh, "Adaptive clonal selection algorithm for solving OPF problem with emission constraints," in *2013 Annual IEEE India Conference (INDICON)*, IEEE, Dec. 2013, pp. 1–6. doi: 10.1109/INDCON.2013.6725969.

[24] B. Haktanirlar Ulutas and S. Kulturel-Konak, "A review of clonal selection algorithm and its applications," *Artif. Intell. Rev.*, vol. 36, no. 2, pp. 117–138, Aug. 2011, doi: 10.1007/s10462-011-9206-1.

[25] J. Greensmith, U. Aickelin, and G. Tedesco, "Information fusion for anomaly detection with the dendritic cell algorithm," *Inf. Fusion*, vol. 11, no. 1, pp. 21–34, Jan. 2010, doi: 10.1016/j.inffus.2009.04.006.

[26] N. Elisa, L. Yang, X. Fu, and N. Naik, "Dendritic Cell Algorithm Enhancement Using Fuzzy Inference System for Network Intrusion Detection," in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, Jun. 2019, pp. 1–6. doi: 10.1109/FUZZ-IEEE.2019.8859006.