Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.079034

ARTICLE

Dendritic Cell Algorithm with Reinforcement Learning for Adaptive Signal Categorization

Yousra Abudaqqa*, Zulaiha Ali Othman and Azuraliza Abu Bakar

Research Center for Artificial Intelligent Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia

*Corresponding Author: Yousra Abudaqqa. Email: yousram.83@gmail.com

Received: 13 January 2026; Accepted: 26 March 2026; Published: 27 May 2026

ABSTRACT: Signal categorization is a critical component of the Dendritic Cell Algorithm (DCA), as it directly influences its anomaly detection capability. Conventional DCA implementations typically rely on heuristic or optimization-based approaches, such as Grouping Particle Swarm Optimization (GPSO), Grouping Genetic Algorithms (GGA), Principal Component Analysis (PCA), and Support Vector Machines (SVM), to determine mappings between input features and the three immunological signal categories: Pathogen-Associated Molecular Patterns (PAMP), Danger Signals (DS), and Safe Signals (SS). These approaches depend heavily on domain expertise and predefined rules, making the resulting signal mappings static and often dataset specific. Consequently, the traditional DCA lacks flexibility across diverse data domains and may fail to capture evolving patterns in complex datasets. To address this limitation, this study integrates Reinforcement Learning (RL) into the DCA framework to develop an adaptive signal categorization mechanism. The proposed RL-DCA model employs a Q-learning agent to dynamically assign features to the three signal categories based on reward feedback derived from classification performance. Through continuous interaction with the environment, the RL agent learns an optimal signal mapping policy that improves the quality of generated signals while reducing reliance on manually defined configurations. Experimental evaluations conducted on nine benchmark datasets from multiple domains demonstrate that the proposed RL-DCA framework consistently outperforms existing DCA variants in terms of anomaly detection accuracy and robustness. The results confirm that reinforcement learning provides an effective mechanism for enabling adaptive and data-driven signal categorization in immune-inspired anomaly detection systems.

KEYWORDS: Dendritic cell algorithm (DCA); reinforcement learning (RL); Q-learning; dynamic signal categorization; anomaly detection

1 Introduction

The natural immune system exhibits remarkable properties such as adaptability, diversity, distributiveness, and robustness, enabling it to effectively recognize and eliminate harmful pathogens. Artificial Immune Systems (AIS) are a family of bio-inspired computational algorithms developed by modeling these biological immune mechanisms and applying them to a wide range of computational problems. From a computational perspective, AIS approaches offer several advantages that make them attractive for solving complex problems in computer science. By incorporating principles inspired by the natural immune system, AIS models often provide unique capabilities such as adaptability, fault tolerance, and robustness [1]. Many AIS algorithms are considered computationally lightweight in classification and anomaly detection tasks when compared with conventional machine learning methods. Furthermore, previous studies have reported that AIS-based

approaches can achieve competitive or superior detection performance in several application domains while maintaining good generalization ability [2].

Among the various AIS algorithms, the Dendritic Cell Algorithm (DCA) is one of the most widely studied models. DCA emulates the behavior of biological dendritic cells (DCs), which play a key role in the innate immune system by detecting potentially harmful signals and coordinating immune responses. Inspired by this biological mechanism, the DCA has been successfully applied to anomaly detection tasks in several domains, including intrusion detection, fault diagnosis, and spam filtering [3]. The algorithm is particularly suitable for binary classification problems, where each input instance is transformed into three biologically inspired signal types: Pathogen-Associated Molecular Patterns (PAMP), Danger Signals (DS), and Safe Signals (SS). These signals are processed together with data identifiers (antigens) through a context assessment mechanism that determines whether a given instance represents normal or anomalous behavior [4].

In general, the DCA consists of four main processing phases: (i) preprocessing and initialization, (ii) detection, (iii) context assessment, and (iv) classification [3,4]. Among these stages, the preprocessing phase plays a critical role because it transforms raw input features into the three signal categories (PAMP, DS, and SS) that drive the entire detection process. Consequently, the effectiveness of the DCA largely depends on the quality of feature selection and the correctness of feature-to-signal mapping. Accurate identification of informative features and appropriate assignment of those features to the corresponding signal categories are essential to ensure that the generated signals are meaningful and discriminative for anomaly detection.

Traditionally, the preprocessing stage of the DCA relies heavily on manual configuration or static feature transformation techniques. Methods such as Principal Component Analysis (PCA) [3], Correlation Coefficient (CC) [4], Information Gain (IG), Rough Set Theory (RST) [5], and Fuzzy Rough Set Theory (FRST) [6] have been widely used to construct reduced feature spaces with lower dimensionality. These techniques typically remove weak or irrelevant attributes based on statistical relevance measures. However, relevance according to these criteria does not necessarily guarantee that a feature belongs to the optimal DCA feature subset, nor does irrelevance imply that a feature is completely unsuitable. As a result, filter-based reduction methods may produce suboptimal signal mappings that negatively affect anomaly classification performance [7].

To further improve preprocessing, several studies have incorporated machine learning and optimization techniques into the DCA framework. For example, Support Vector Machines (SVM) [8] and K-Nearest Neighbors (KNN) [9] have been used to assist feature selection or signal generation. In addition, optimization-based approaches such as Grouping Particle Swarm Optimization (GPSO) [10] and Grouping Genetic Algorithms (GGA) [11] formulate signal categorization as a combinatorial grouping problem. In these approaches, input features are divided into mutually disjoint subsets representing PAMP, DS, SS, and unused features. These techniques partially automate the signal categorization process and reduce manual intervention. However, the resulting mappings remain static, meaning that once the optimization process finishes, the feature-to-signal assignments remain fixed.

A major limitation of evolutionary optimization approaches such as PSO-DCA and GA-DCA is their limited generalization capability. Because these algorithms optimize feature groupings based on a specific dataset, the resulting mappings often reflect dataset-specific statistical patterns rather than generalizable relationships. Consequently, the learned mappings may perform poorly when applied to new datasets or different domains [12]. In addition, PSO is highly sensitive to parameter configuration and does not incorporate the semantic meaning of the DCA signal categories. Similarly, GA-based methods may suffer from instability due to stochastic operations such as mutation and crossover. As a result, these optimization methods typically

require re-optimization for each new dataset, which limits their applicability in multi-domain anomaly detection tasks where adaptability and robustness are required.

Another limitation of the traditional DCA framework is the reliance on static and manually defined signal mappings. In many implementations, users must assign features to PAMP, DS, and SS based on intuition or domain knowledge [13]. Once defined, these mappings remain fixed and cannot adapt to evolving data distributions or changing anomaly patterns. This rigid structure limits the ability of the DCA to respond to new or previously unseen data characteristics, thereby reducing its effectiveness in dynamic environments.

To address these challenges, this study proposes a novel adaptive preprocessing framework called Reinforcement Learning-based DCA (RL-DCA). The proposed approach reformulates the feature-to-signal mapping process as a reinforcement learning (RL) task [14,15]. Specifically, the RL-DCA model employs Q-learning [16] to iteratively optimize the assignment of features to the three signal categories (PAMP, DS, and SS) based on reward feedback derived from classification performance. Unlike static signal mappings, the RL agent continuously updates its signal categorization policy through interaction with the environment. This adaptive mechanism enables RL-DCA to automatically adjust signal mappings as data characteristics change, thereby reducing reliance on manual configuration.

Such adaptability is particularly important in domains such as cybersecurity, financial fraud detection, and behavioral analytics, where anomaly patterns evolve over time and static preprocessing methods may quickly become ineffective. By incorporating a self-learning mechanism into the preprocessing phase, RL-DCA can dynamically adapt to changing data distributions and maintain stable anomaly detection performance across heterogeneous datasets.

Motivated by these challenges, this study addresses the following research question:

How can the preprocessing phase of the DCA be made adaptive, learning-driven, and generalizable without relying on manual configuration or static optimization strategies?

To answer this question, the main contributions of this study are summarized as follows:

1. Adaptive Signal Categorization: An RL-based signal categorization mechanism (RL-DCA) is introduced to automatically learn feature-to-signal mappings in the preprocessing stage of the DCA.
2. Comprehensive Evaluation: The proposed RL-DCA model is evaluated on nine benchmark datasets from multiple domains and compared with several existing DCA variants, including GGA-DCA, PSO-DCA, PCA-DCA, and SVM-DCA.
3. Computational Analysis: The time complexity and runtime behavior of the RL-DCA framework are analyzed and compared with existing DCA-based models.

The remainder of this paper is organized as follows. Section 2 presents an overview of the Dendritic Cell Algorithm and its computational principles. Section 3 introduces the reinforcement learning framework based on Q-learning. Section 4 reviews related work on DCA preprocessing and signal categorization. Section 5 describes the proposed RL-DCA model. Section 6 presents the experimental setup and datasets used in the evaluation. Section 7 reports and analyzes the experimental results. Section 8 investigates the sensitivity of the MCAV threshold. Section 9 discusses the implications and limitations of the proposed approach. Finally, Section 10 concludes the paper and outlines directions for future research.

2 Overviews of DCA

The DCA is an AIS technique inspired by the behaviour of biological DCs in the innate immune system. In immunology, DCs act as professional antigen-presenting cells with the ability to ingest, process, and present antigens to T-cells. Their behaviour is regulated by environmental signals: in the presence of pathogenic or danger cues, DCs transition to a mature state, triggering an immune response; in contrast,

exposure primarily to safe signals leads to a semi-mature state that promotes immune tolerance. This dual-state mechanism is the biological foundation for distinguishing between normal and anomalous patterns [3].

For clarity, the key variables and notations used throughout this study are summarized here. The DCA operates using three immunological signal categories: Pathogen-Associated Molecular Patterns (PAMP), Danger Signals (DS), and Safe Signals (SS). During the detection phase, each dendritic cell accumulates three output signals: the costimulatory signal (Csm), the mature output signal (MAT), and the semi-mature output signal (SEMI). The migration threshold (MT) determines when a dendritic cell stops sampling antigens and performs context assessment. The final classification decision is derived using the Mature Context Antigen Value (MCAV), which measures the proportion of mature contexts associated with each antigen. In the reinforcement learning component, α denotes the learning rate, γ represents the discount factor, and ε controls the exploration rate in the ε-greedy policy. These parameters regulate how the RL agent updates its signal assignment strategy during training.

2.1 Basic Computational Definitions

In the DCA, each data instance is represented using two components: antigens and signals.

Definition 1 An antigen is defined as $Ag = \langle e, t \rangle$, where e is the unique identifier of a data instance, and t is the timestamp: *Antigens represent what is being classified.*

Definition 2 Signals are defined as follows: *Each data instance is transformed into a three-dimensional real-valued signal vector $Signal = (PAMP, DS, SS)$, where: PAMP is a strong an indicator of anomaly, DS is a moderate indicator of anomaly, SS is an indicator of normality.*

Definition 3 DCs: *A single artificial dendritic cell is defined as $DC = Ags \times Signals \times T$. where: Ags is a set of antigens sampled by the DC, $Signals$ is the cumulative signal profile of that DC, and T is the migration threshold that determines when the DC should stop sampling and assess the environment. The DCA operates with a population of DCs, each contributing to the final anomaly decision.*

2.2 Overall Structure of the DCA

Following Greensmith & Aickelin's abstraction, the DCA can be written as a function: H = A × S × N, where: A is the antigen set, S is the transformed signal set, and N is the population of DCs. As formalized in Algorithm 1 and presented in Fig. 1, The DCA algorithm consists of four main phases include: (1) Pre-processing and Initialization, (2) Detection Phase, (3) Context Assessment, and (4) Classification Phase. Each phase contributes essential mechanisms for transforming input data into anomaly labels.

1. **Pre-processing and Initialization Phase:** The first phase prepares the dataset for immune-inspired processing. Each data instance is transformed into two components: an antigen (identifier) and a signal vector consisting of PAMP, DS, and SS signals. These signals are derived through feature selection and domain mapping, enabling the algorithm to convert high-dimensional data into the three-signal structure required by DCA. A population of artificial DCs is then initialized. Each DC is assigned: a random migration threshold T, empty antigen storage, and zero cumulative values for the output signals: Costimulatory molecule (Csm), Semi-mature (SEMI), and Mature (MAT). This initialization prepares the DCs to begin sampling and accumulating signals from the environment.
2. **Detection Phase:** The detection phase is the core operational stage of the DCA. Each dendritic cell continuously samples antigens and combines the input signals with a predefined weight matrix, shown in Table 1, to produce three interim outputs: Costimulatory signal (Csm), Semi-mature (SEMI), and Mature (MAT).

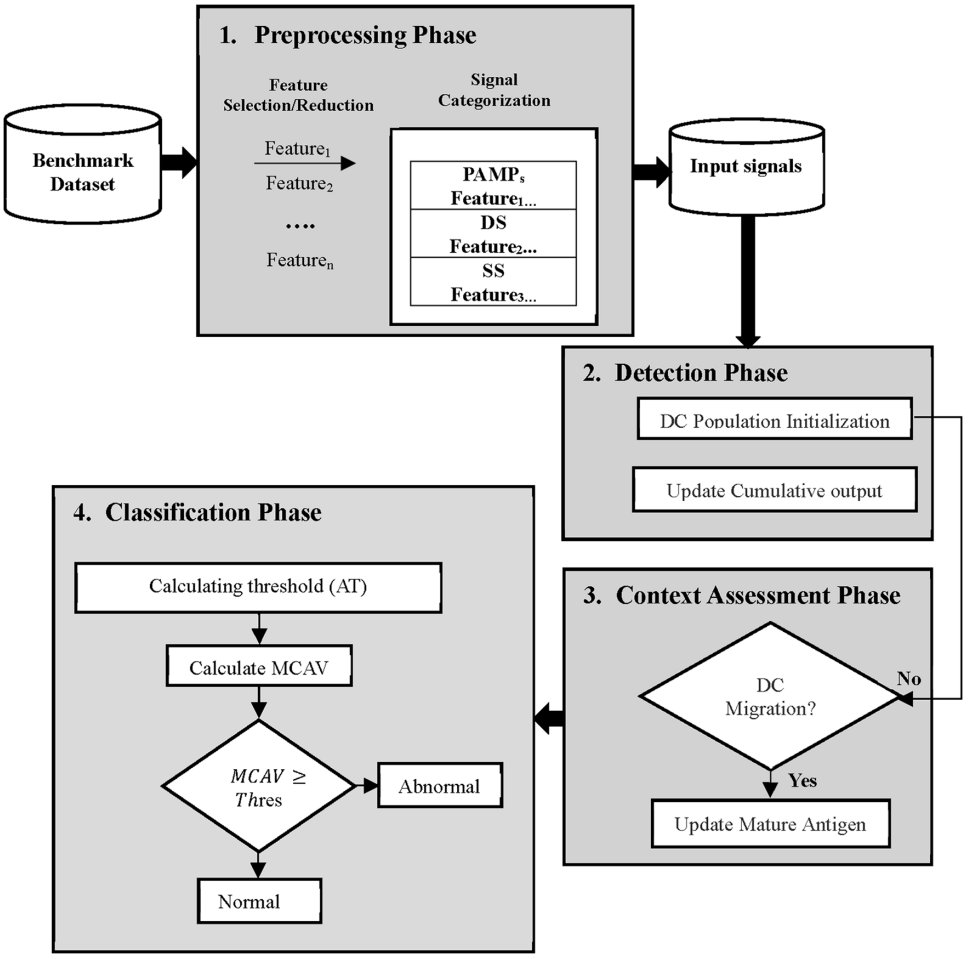

Figure 1: The standard DCA model.

Algorithm 1: The dendritic cell algorithm

Input: Signals from all categories and antigens: SS, PAMP, DS, and antigens.
Output: Antigen context values (binary classification: 0 = normal, 1 = abnormal).

for each DC **do**
/* Preprocessing and Initialization Phase */
1: initialize DC parameters.
2: **end for**
3: **while** CSM < MT **do**
/* Detection Phase */
4: get-antigens();
5: get-signals();
6: calculate-inter();
7: update-cumul();
8: **end while**
/* Context Assessment Phase */
9: **if** smDC > mDC **then**

(Continued)

Algorithm 1 (continued)

```
10:          cell-context = 0;
11:    else
12:          cell-context = 1;
13:    end if
       /* Classification Phase */
14:    for each antigen presented by the DC do
15:          if cell-context = 1 then
16:                Nb-mature ← Nb-mature + 1;
17:          end if
18:    end
19:    for each antigen do
20:          MCAV = Nb-mature/Nb-antigen;
21:    end
```

Table 1: Default weight matrix [5].

Signal Type	Csm	Mature	Semi-Mature
PAMP	2	1	2
DS	0	0	3
SS	2	1	–1

The combination of signals is computed using the weighted linear transformation shown in Eq. (1).

Let P_i, SS_i, and DS_i denote the sampled PAMP, safe, and danger signals, respectively, and let w_P, w_{SS}, and w_{DS} represent their corresponding weights.

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

Every DC continues accumulating the three output values until the Csm value exceeds the migration threshold T. This mechanism prevents DCs from having an excessively long lifespan.

1. **Context Assessment Phase:** Once a DC's accumulated Csm surpasses its migration threshold, it stops sampling new antigens and enters the context assessment phase. If the Mature output value is greater than the Semi-mature output value, the DC indicates that the sampled antigens are associated with an abnormal context. Conversely, if the Semi-mature output is higher, the DC marks its sampled antigens as normal. This biological analogy reflects the natural behaviour of dendritic cells, where mature DCs stimulate immune activation, while semi-mature DCs promote tolerance.
2. **Classification Phase:** The final step of the algorithm aggregates the decisions made by all migrated dendritic cells. Since each antigen may be sampled by multiple DCs, the algorithm computes the Mature Context Antigen Value (MCAV) for each antigen to measure the degree of abnormality. MCAV is defined in Eq. (2):

$$MCAV = \frac{\text{matured-cells}}{\text{presented-cells}} \tag{2}$$

A value close to 1 indicates that most DCs handling that antigen matured before migration signifying a high likelihood of anomaly. A value close to 0 reflects normal behaviour. The MCAV is compared against

an anomaly threshold, which may be determined automatically from the dataset or defined by the user. If $MCAV \geq \theta$, the antigen is classified as anomalous. Otherwise, it is classified as normal.

3 Reinforcement Learning Using Q-Learning

Reinforcement Learning (RL) [6] is a sequential decision-making paradigm in which an agent learns an optimal policy through direct interaction with an environment by maximizing cumulative reward. In the proposed RL-DCA framework, RL is not employed as a conventional classifier. Instead, it serves as a search and optimization mechanism to dynamically learn feature to signal mappings for the DCA. This design directly addresses a key limitation of classical DCA, namely its reliance on manually defined and static signal categorization rules that are often domain dependent and difficult to generalize.

Among various RL approaches, Q-learning is adopted due to its simplicity, model-free nature, and proven convergence properties. Q-learning learns an action–value function $Q(s, a)$, which estimates the expected long-term reward of action a in state s and following the optimal policy thereafter. In the context of RL-DCA, the environment state s is derived from a compact representation of the input features, while the action space consists of three biologically inspired signal categories: PAMP, DS, and SS. Each action corresponds to assigning a particular signal type to the current data instance.

At each interaction step, the RL agent selects an action using an ε-greedy strategy, which balances exploration of alternative signal assignments and exploitation of learned knowledge. The reward function is label guided and designed to reinforce biologically meaningful behavior: assigning PAMP to abnormal samples or SS to normal samples yields a positive reward, while incorrect assignments are penalized. This reward structure allows the agent to gradually learn which feature patterns correspond to each signal category without relying on expert-defined thresholds or fixed heuristics. The Q-values are updated iteratively using the Bellman optimality equation, shown as Eq. (3):

$$\mathrm{Q(s,a)} \leftarrow \mathrm{Q(s,a)} + \alpha[\mathrm{r} + \gamma \max \mathrm{Q(s', a')} - \mathrm{Q(s,a)}] \tag{3}$$

where α is the learning rate, γ is the discount factor, r denotes the immediate reward, and s' represents the next observed state. Through repeated updates, the Q-table gradually converges toward an optimal signal mapping policy that reflects the relationship between feature patterns and immune signal categories. The learned signal vectors are subsequently passed to the DCA, which performs the final immune-inspired aggregation and decision-making process. In this way, Q-learning enhances the adaptability of DCA while preserving its biological interpretation and anomaly detection mechanism.

In the proposed RL-DCA framework, reinforcement learning is used as a label-guided optimization mechanism for signal categorization. Because the reward function uses ground-truth class labels during training, this stage introduces a supervised learning component. However, RL is applied only to optimize signal generation before the standard DCA processing stage, while the final anomaly classification is still performed by the original DCA context assessment and MCAV mechanism.

4 Related Work

The preprocessing phase of DCA, particularly feature selection or feature reduction, and signal categorization, plays a crucial role in its anomaly detection performance. This phase determines how raw input features are transformed into the three core signal types: PAMP, DS, and SS [3].

In classical DCA implementations, preprocessing is typically performed manually using rule-based or statistically driven mappings informed by domain expertise. Practitioners select relevant features and assign them to signal categories based on intuition, prior knowledge, or predefined thresholds [3]. Although

this approach is simple and interpretable, it introduces subjectivity and rigidity, which limit scalability and adaptability, especially in high-dimensional, noisy, or dynamic environments.

To mitigate the limitations of manual preprocessing, several feature reduction techniques have been incorporated into the DCA framework. Principal Component Analysis (PCA) [8] is among the most widely used methods, where original features are transformed into a smaller set of uncorrelated components before being mapped to DCA signals. PCA improves computational efficiency and reduces redundancy; however, it often sacrifices semantic interpretability, making it difficult to trace how individual features influence signal behavior [9].

Other statistical feature selection methods, such as Information Gain, correlation coefficients, and mutual information, have also been employed to rank and select features prior to signal mapping [10]. While effective in certain scenarios, these approaches generally assume linear relationships and often fail to capture complex feature interactions, reducing their robustness in noisy or non-linear datasets [11].

More recent studies have explored automated feature reduction techniques to further enhance the DCA preprocessing phase. Kernel Principal Component Analysis (KPCA) has been proposed as a nonlinear extension of PCA, enabling projection into kernel-induced feature spaces that better preserve complex relationships and filter noise prior to DCA execution. Autoencoders (AE) have also been adopted to learn compact latent representations through unsupervised neural reconstruction, allowing nonlinear feature compression before mapping reduced features into DCA signals. Hybrid approaches, such as AEkPCA, combine AE-based representation learning with KPCA refinement to further decorrelate and enhance latent features. These methods have demonstrated improved anomaly detection performance across multiple domains. Despite their effectiveness, KPCA-, AE-, and AEkPCA-based pipelines introduce additional computational overhead, hyperparameter sensitivity, and static transformation stages that operate independently of the DCA decision process. Consequently, while these approaches reduce manual intervention, they remain external preprocessing modules rather than adaptive mechanisms integrated within the DCA framework itself [12].

Several studies have focused specifically on improving signal categorization in DCA. Mean Squared Error (MSE) and signal sensitivity measures have been used to evaluate the quality of feature-to-signal assignments. Rough Set Theory (RST) has also been applied to identify minimal feature subsets (reducts) that preserve classification performance [13]. The QuickReduct algorithm, for example, derives efficient feature sets while maintaining DCA accuracy. To handle uncertainty in feature importance, Fuzzy Rough Set Theory (FRST) has been introduced, enabling soft boundaries during categorization [13]. Although these techniques improve robustness, their configurations remain static and highly dependent on dataset characteristics.

To further enhance input signal generation, machine learning classifiers have been integrated with the traditional DCA. In SVM-DCA, Support Vector Machines are used to compute sparse weight matrices, where attributes with larger weights are retained and mapped to DCA signals [14]. Similarly, K-Nearest Neighbors (KNN) has been employed to filter attributes based on pairwise instance differences, forming hybrid KNN-DCA models.

Metaheuristic optimization techniques have also been explored to automate signal mapping. Genetic Algorithms (GA) evolve feature-to-signal assignments using mutation and crossover operations, optimizing classification accuracy [15,16]. Particle Swarm Optimization (PSO) and Genetic Grouping Algorithms (GGA) further frame signal categorization as a clustering task, grouping features into PAMP, DS, SS, and unassigned categories [14,17]. These methods reduce manual configuration but typically converge to fixed mappings. Additional studies have incorporated Bayesian Optimization with resource-efficient strategies such as Hyperband to fine-tune signal fusion parameters [18]. Although these methods enhance discriminative power, they still produce static mappings that do not adapt after training.

A critical limitation across existing DCA-based methods is the lack of adaptability in signal categorization. Most approaches generate fixed feature-to-signal mappings that cannot respond to evolving data distributions, class imbalance, or noise. For example, PCA may perform well on one dataset but poorly on another, while GA- and PSO-based methods are often sensitive to parameter tuning and dataset complexity [19].

Although the DCA is bio-inspired and widely used for anomaly detection [20], it also suffers from similar weaknesses [21]. In its standard form, the DCA relies on experience-based parameter settings and lacks a training phase, resulting in reduced categorization accuracy when handling large-scale or high-dimensional datasets. This rigid design limits robustness under real-world conditions.

Recent advances in artificial intelligence have significantly influenced the development of zero-day and anomaly detection systems. A comprehensive review by Yee et al. [22] highlights that AI-based detection methods can be broadly categorized into machine learning based, deep learning based, hybrid, and anomaly based approaches. While many supervised and deep learning models achieve high detection accuracy, the review emphasizes recurring challenges such as limited adaptability, dependence on labeled data, sensitivity to data distribution shifts, and difficulty handling unseen attack patterns. These limitations indicate that static learning mechanisms may struggle to maintain performance in evolving threat environments.

Similarly, Dai et al. [23] propose a hybrid intrusion detection framework that integrates an autoencoder with Random Forest and XGBoost classifiers to improve detection of zero-day attacks in unseen data. Their results demonstrate that incorporating anomaly detection improves generalization within the same dataset distribution. However, the model structure remains largely dependent on supervised classifiers after anomaly filtering, and adaptation does not occur within the internal feature transformation process itself. These findings collectively suggest that improving generalization requires not only anomaly aware modeling but also mechanisms that allow detection systems to adapt dynamically during operation.

Inspired by advances in adaptive optimization, recent studies have investigated Reinforcement Learning (RL) as a mechanism for enabling self-learning and adaptability in computational models. Liu et al. [24] applied deep RL for feature selection in high-dimensional domains, outperforming static filter and wrapper methods. Other works have used RL to optimize decision thresholds in network anomaly detection [25], adapt detection parameters in artificial immune systems [26], and control parameter updates in swarm intelligence algorithms [27]. These studies consistently demonstrate that RL enables models to adjust their behavior based on environmental feedback rather than relying on fixed configurations.

Motivated by both the limitations identified in recent AI-based zero-day detection frameworks and the adaptive capability demonstrated by RL-driven models, this study introduces reinforcement learning into the Dendritic Cell Algorithm (DCA) to enhance its signal categorization mechanism. Unlike static or handcrafted mappings, RL frames signal categorization as a sequential decision-making process guided by feedback from classification performance.

Specifically, this work replaces the static signal categorization mechanism with a dynamic RL-based strategy using Q-learning [28]. Through continuous interaction with the environment, the RL agent evaluates outcomes and iteratively updates its signal assignment policy. This design enables the DCA to self-adjust its internal signal transformation process in real time, thereby improving adaptability, generalization capability, and robustness across diverse benchmark datasets. A comparative overview of existing DCA preprocessing methods, including their advantages, limitations, and adaptability, is presented in Table 2.

Table 2: Summary of DCA preprocessing methods in terms of advantages, limitations, and adaptability. The proposed RL-DCA is the only method that supports dynamic and feedback-driven signal categorization.

Method	Application	Advantages	Limitations	Adaptability
Manual Mapping [10]	Signal Categorization	Simple and interpretable, easy to implement	Subjective, non-scalable, domain-specific	No
PCA [29]	Feature Reduction	Removes redundancy, improves computational efficiency	Loses interpretability, assumes linearity	No
Information Gain, Correlation, MI [30]	Feature Selection	Fast feature selection, highlights relevance	Ignores feature interactions, not robust to noise	No
RST [31]	Feature Selection	Identifies minimal feature subsets (reducts)	Requires discretization, performance varies across datasets	No
FRST [13]	Feature Selection	Models' uncertainty, flexible signal boundaries	Computationally complex, static configuration	No
GA [16]	Feature Selection	Explores large search spaces, adaptable to objective functions	Parameter sensitivity, performance inconsistency	Limited
GGA [17]	Signal Categorization	Automates signal group assignment, retains interpretability	Produces static mappings, lacks online adaptability	Limited
GPSO [14]	Signal Categorization	Fast convergence, fewer parameters than GA, global search capability	Susceptible to local optima, static output post-training	Limited
Bayesian Optimization + Hyperband [18]	Signal Categorization	Efficient exploration of parameter space, improves signal fusion performance	Resource-intensive, mappings fixed post-training	No
Proposed RL-DCA	Signal Categorization	Learns from classification feedback, dynamically adapts signal mappings	Requires training and reward shaping	Yes (Fully Adaptive)

5 The Proposed Model: RL-DCA

5.1 Model Overview

This study aims to transform the traditional feature selection and signal categorization procedures adopted by existing DCA-based approaches (e.g., PCA-DCA, GPSO-DCA, and GGA-DCA) into a dynamic and self-learning mechanism using RL with the Q-learning algorithm. Instead of relying on predefined or population-based feature groupings, the proposed RL-DCA framework enables the feature-to-signal

assignment process to be learned adaptively through continuous interaction with the environment. In RL-DCA, an RL agent dynamically determines how input

Features should be mapped to the three primary immunological signal categories: PAMP, DS, and SS. Features that do not contribute meaningfully to these signal categories are implicitly excluded from further signal processing. Through this adaptive learning mechanism, RL replaces static signal categorization rules with a data driven optimization process.

A novel hybrid scheme, termed RL-DCA, is introduced by integrating RL-based dynamic signal transformation with the DCA, as illustrated in Fig. 2. RL-DCA consists of three main components: the search space, the search engine, and the evaluation method. The search space represents all possible feature to signal assignments, the RL agent serves as the search engine that optimizes this mapping, and the DCA functions as the evaluation method by assessing classification performance.

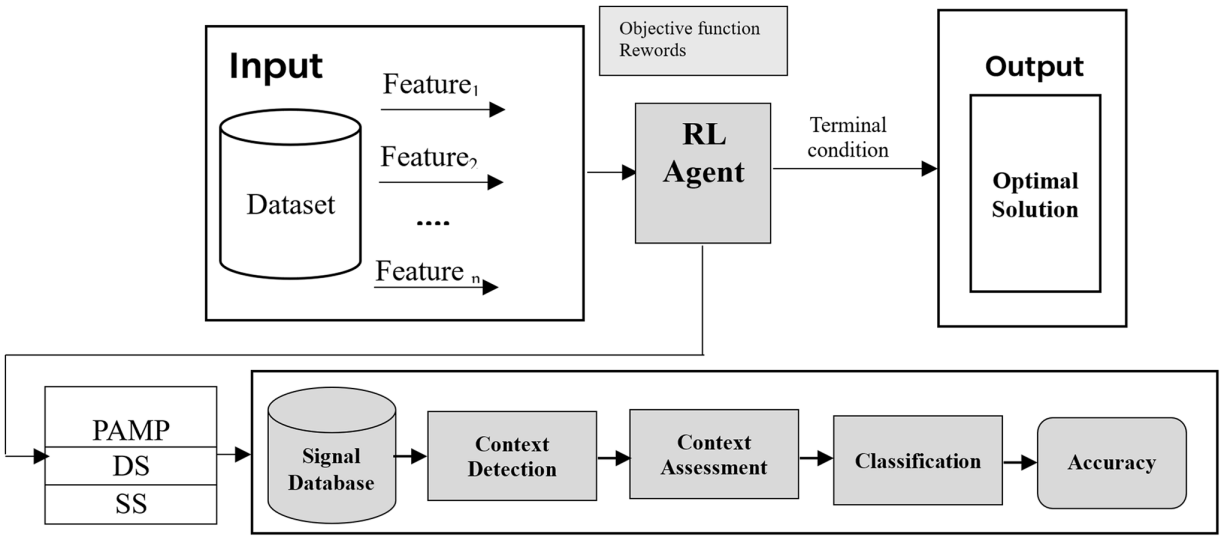

Figure 2: The model of RL-DCA.

5.2 Search Space

In RL-DCA, the search space is defined as the set of all possible mappings between input features and immunological signal categories. As shown in Fig. 3, The signal categorization process establishes a mapping relationship between features and signal types, resulting in a three-dimensional signal representation, as expressed in Eq. (4):

$$\{Feature_i, Feature_j, \ldots\} \rightarrow \{PAMP, DS, SS\}. \tag{4}$$

Each mapping in the search space represents a potential solution for generating input signals for the DCA. Through interaction with the environment and reward feedback, the RL agent explores this search space and progressively refines the feature to signal assignment strategy.

5.3 Search Method: Reinforcement Learning Using Q-Learning

This study employs reinforcement learning with the Q-learning algorithm as the search engine to identify optimal feature to signal mappings. Q-learning is selected due to its model free nature and its ability to learn optimal policies through iterative interaction with the environment.

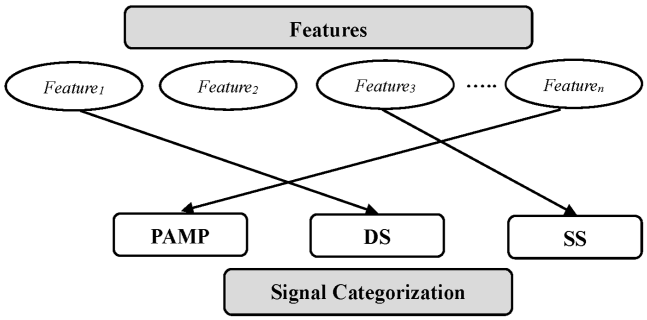

Figure 3: Steps of feature selection and signal categorization.

Step 1: Reinforcement Learning Based Signal Categorization

In the proposed RL-DCA framework, reinforcement learning is employed to dynamically determine the assignment of input instances to immunological signal categories used by the DCA. Unlike conventional approaches that rely on manually predefined feature-to-signal mappings, the RL agent learns an adaptive signal assignment policy through interaction with the environment. Each state represents a compact representation of the input instance derived from the feature space, while each action corresponds to assigning the instance to one of the immunological signal categories defined in Eq. (1). The Q-learning algorithm iteratively updates the signal assignment policy based on reward feedback derived from classification consistency, allowing the agent to gradually learn which signal categories best represent the underlying data patterns.

Step 2: RL State Representation via Random Projection and Discretization

Let $x_i \in \mathbb{R}^d$ denote the normalized feature vector of the i-th input instance after preprocessing, where d represents the number of features. Directly using x_i as a state in tabular Q-learning is infeasible because the number of possible states grows exponentially with the dimensionality of the feature space. To obtain a compact yet informative state representation while still incorporating information from all input features, a Random Projection (RP) [32] transformation is employed. Specifically, each feature vector is projected into a two-dimensional embedding space using a randomly generated projection matrix as shown in Eq. (5):

$$z_i = R^T x_i \tag{5}$$

where $R \in \mathbb{R}^{d\times 2}$ is a fixed random matrix, whose elements are sampled from a standard normal distribution $r_{jk} \sim \mathcal{N}(0,1)$. A fixed random seed is used to ensure reproducibility. The resulting projected components are therefore linear combinations of all original features as shown in Eq. (6):

$$z_{i1} = \sum_{j=1}^{d} r_{j1} x_{ij}, z_{i2} = \sum_{j=1}^{d} r_{j2} x_{ij} \tag{6}$$

Thus, although the RL agent operates in a two-dimensional state space, the representation implicitly incorporates information from all original features through the projection transformation. This design follows the Johnson–Linden Strauss principle, which states that random projections approximately preserve pairwise distances when reducing dimensionality.

To stabilize tabular learning and limit the number of possible states, the projected vector $z_i = [z_{i1}, z_{i2}]^T$ is discretized using a binning strategy. After discretization, the RL state is represented as a two-dimensional tuple $s_i = (b_{i1}, b_{i2})$ where b_{i1} and b_{i2} denote the discretized bin indices corresponding to the two projected feature components. This compact state representation enables the tabular Q-learning agent to operate within a finite state space while still capturing information from the original high-dimensional feature set.

First, each component is clipped to a bounded interval: $[z_{\min}, z_{\max}]$. The discretized state is then computed as: $s_i = (b_{i1}, b_{i2})$, where shown in Eq. (7)

$$b_{ik} = \left\lfloor \frac{\text{clip}\,(z_{ik}, z_{\min}, z_{\max}) - z_{\min}}{z_{\max} - z_{\min}} \times B \right\rfloor, k \in \{0, 1\} \tag{7}$$

and B denotes the number of discretization bins. This discretization process ensures that the RL agent operates within a finite state space while still reflecting patterns from the full feature space through the random projection.

Step 3: Signal Mapping via Q-Learning

The signal categorization process is formulated as a Markov Decision Process (MDP) [33] defined by the state space S, action space A, reward function R, and transition dynamics. The RL agent follows a Q-learning strategy using an $\boldsymbol{\varepsilon}$-greedy policy to balance exploration and exploitation. At each step, the agent observes the current state s_i and selects.

The action space consists of three possible signal assignments defined as $A = \{0, 1, 2\}$, where action 0 assigns the PAMP signal, action 1 assigns the DS, and action 2 assigns the SS. As shown in Eq. (8)

$$A = \{0, 1, 2\}, \tag{8}$$

where:

- 0: assign PAMP
- 1: assign DS
- 2: assign SS

The reward function evaluates the correctness of the signal assignment relative to the class label of the instance. A positive reward is given when anomalous instances are mapped to danger-related signals and normal instances are mapped to safe signals. Formally, the reward function is defined as shown in Eq. (9):

$$r = \begin{cases} +1, & y_i = 1 \text{ and } a \in \{0, 1, 2\} \\ +1, & y_i = 0 \text{ and } a = 2 \\ -1, & \text{otherwise} \end{cases} \tag{9}$$

The Q-value function is iteratively updated according to the Bellman optimality equation, as shown in Eq. (10):

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a' \in A} Q\,(s', a') - Q\,(s, a) \right] \tag{10}$$

where

- α is the learning rate
- γ is the discount factor
- r is the immediate reward
- s' is the next state.

Fig. 4 illustrates the architecture of the Q-learning mechanism used for adaptive signal mapping in the proposed RL-DCA framework. The figure shows how the Q-learning update rule iteratively refines the signal mapping policy based on reward feedback, enabling the agent to dynamically learn appropriate signal assignments.

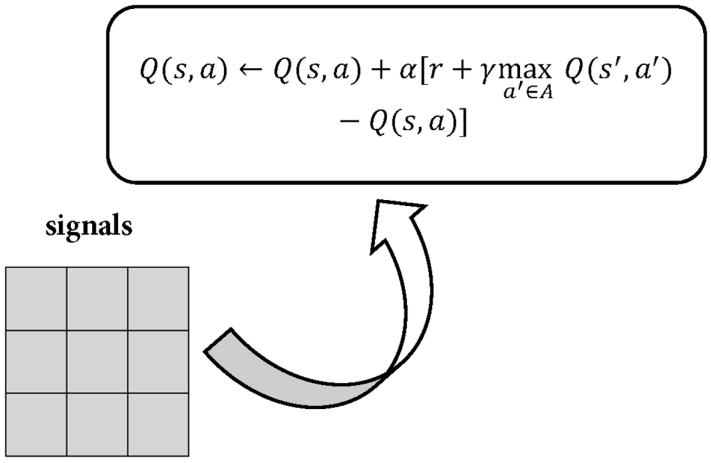

Figure 4: Architecture of the Q-learning mechanism for adaptive signal mapping in the RL-DCA framework.

Through repeated interactions with the environment, the Q-table gradually converges toward an adaptive signal mapping policy.

Step 4: Signal Vector Construction

Based on the action selected by the RL agent, a three-dimensional signal vector is constructed for each instance as shown in Eq. (11):

$$Signal_i = [PAMP,\ DS,\ SS] \tag{11}$$

Only the signal component corresponding to the selected action is populated, while the remaining components are set to zero. As shown in Eq. (12)

$$Signal_i = \begin{cases} [v_i, 0, 0], & a = 0 \\ [0, v_i, 0], & a = 1 \\ [0, 0, v_i], & a = 2 \end{cases} \tag{12}$$

where v_i denotes the signal magnitude extracted from the corresponding feature component of the normalized input vector. The resulting signal vectors form the input signal stream for the subsequent DCA processing stage, where immune-inspired signal aggregation and context evaluation are performed to produce the final anomaly classification.

Fig. 5 illustrates the distribution of the immunological signals generated by the proposed RL-based signal categorization mechanism for the SP dataset. The figure presents the three signal types used in the DCA framework. The horizontal axis represents the sample index, while the vertical axis indicates the magnitude of the corresponding signal assigned by the reinforcement learning agent. As observed in the figure, the RL agent dynamically generates signal values according to the learned feature to signal mapping policy obtained through the Q-learning process. The PAMP signal exhibits several peaks across the dataset, indicating regions where abnormal behavioral patterns are detected. The DS appears sparsely with distinct spikes, highlighting instances that are strongly associated with potential anomalies. In contrast, the SS is more broadly distributed across the dataset and dominates in regions corresponding to normal instances. These RL-generated signals collectively form the three-dimensional signal vector defined in Eqs. (11) and (12) and serve as the input signal stream for the subsequent DCA processing phase. Through this adaptive signal generation process, the proposed RL-DCA framework replaces the static signal categorization used in traditional DCA variants with a data driven mechanism that dynamically learns informative signal representations from the dataset.

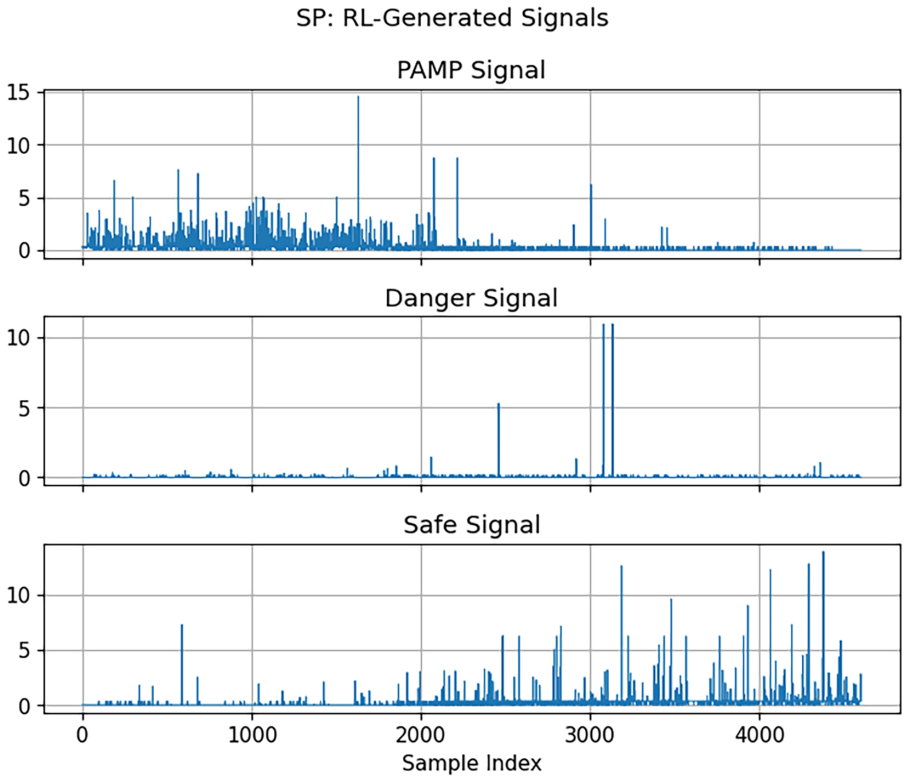

Figure 5: Distribution of signals generated by the proposed RL-DCA method for the SP dataset.

Step 5: Algorithm of RL-DCA

Algorithm 2 summarizes the complete RL-DCA search procedure. The algorithm integrates Q-learning as a search and optimization mechanism for adaptive signal categorization. For each input instance, a compact state representation defined in Eq. (3) is constructed, and an ε-greedy policy is employed to select actions. The Q-values are updated using the Bellman equation in Eq. (4), and the resulting signal vectors defined in Eqs. (5) and (6) are provided as input to the DCA. In this way, reinforcement learning does not replace the dendritic cell algorithm; instead, it enhances its adaptability by enabling self-learning signal generation, thereby reducing reliance on static, expert-defined signal categorization rules and improving robustness across heterogeneous datasets.

Algorithm 2: RL-based signal mapping

Input:

X ∈ R^(N × d) standardized feature matrix
y ∈ [1]^N labels (1 = anomaly, 0 = normal)
α, γ, ε Q-learning parameters
B number of discretization bins
clip_min, clip_max clipping range
n_actions = 3 actions: 0 = PAMP, 1 = DS, 2 = SS

Output:

X_RL ∈ R^(N × 3) signal vectors [PAMP, DS, SS]

1: Begin

(Continued)

Algorithm 2 (continued)

```
Initialize random seed
Sample R ∈ R^(d × 2) with R [j, k] ~ N(0, 1)
Z ← X · R
Initialize Q-table Q [s] = [0, 0, 0]
X_RL ← empty list
For i = 1 to N do
    z ← Z [i]
    z_clip ← clip(z, clip_min, clip_max)
    b1 ← floor((z_clip [1] − clip_min)/(clip_max − clip_min) × B)
    b2 ← floor((z_clip [2] − clip_min)/(clip_max − clip_min) × B)
    b1 ← clip(b1, 0, B − 1)
    b2 ← clip(b2, 0, B − 1)
    s ← (b1, b2)
    If s not in Q then
        Q [s] ← [0, 0, 0]
    End if
    u ← rand(0, 1)
    If u < ε then
        a ← random action
        Else
        a ← argmax_a′ Q [s] [a′]
        End if
        If y [i] = 1 then
            If a ∈ [1] then r ← +1 else r ← −1
        Else
            If a = 2 then r ← +1 else r ← −1
        End if
        s_next ← s
        If s_next not in Q then
            Q [s_next] ← [0, 0, 0]
        End if
        Q [s] [a] ← Q [s] [a] + α (r + γ max_a′ Q [s_next] [a′] − Q [s] [a])
        signal ← [0, 0, 0]
        If a = 0 then
            signal [0] ← X [i] [0]
        Else if a = 1 then
            signal [1] ← X [i] [1]
        Else
            signal [2] ← X [i] [0]
        End if
        Append signal to X_RL
    End for
    Return X_RL
    End
```

6 Experimentations

A Data Sets

To evaluate the performance and generalizability of the proposed RL-DCA model, nine publicly available datasets were selected from reputable sources, including the UCI Machine Learning Repository [34], KEEL [35] and established cybersecurity research platforms [36,37]. These datasets represent a diverse set of binary classification tasks, varying significantly in feature (ranging from 3 to 85 features), sample sizes, and class imbalance levels. The datasets are collected from a variety of domains, including healthcare, finance, social behaviour, spam detection, and network intrusion detection. which allows a comprehensive evaluation of RL-DCA across multiple real-world contexts. Table 3 provides the details of each dataset, including the number of features, the total number of samples, and the ratio of class imbalance. To ensure compatibility with the RL-based signal transformation process, all categorical attributes were numerically encoded using label encoding. Numerical features were standardized via Z-score normalization to align data scales and enhance the convergence behavior of the learning algorithm. Data cleaning procedures included the removal of features with over 40% missing values and attributes with zero variance (i.e., those containing a single unique value). Outliers with a Z-score greater than 2.5 were either filtered or imputed using the feature mean, based on the Eq. (13):

$$z_{i,j} = (x_{i,j} - \mu_i)/\sigma_j \tag{13}$$

where $x_{i,j}$ denotes the value of the i_{th} instance for the j_{th} feature, μ_i is the mean, and σ_j is the standard deviation of that feature. This preprocessing ensures that the selected datasets are standardized, clean, and suitable for evaluating the adaptability and robustness of the RL-DCA algorithm across both general classification tasks and real-world anomaly detection scenarios.

Table 3: Description of datasets used for RL-DCA evaluation.

Dataset	Features	Samples	Domain	IR (Imbalance Rate)
Cervical Cancer Behavioral Risk (CCBR)	19	72	Health (UCI)	~1.6
German Credit Data (GCD)	20	1000	Finance (UCI)	~1.2
Titanic Survival (TITANIC)	3	2201	General (Keel)	~1.58
Divorce Predictors (DP)	54	170	Social (UCI)	~1.43
Spambase (SP)	57	4601	Text/Spam	~1.54
Cheese Spectroscopy (SPCHEES)	36	3196	Food science	~1.2
NSL-KDD Intrusion Detection	41	125,973	Cyber Security	~5.3
UNSW15 Intrusion Detection	49	175,341	Cyber Security	~9.4
Insurance Company Benchmark (COIL2000)	85	9822	Insurance/Finance	~5.9

B Experiment Setup

In this study, a set of experiments was conducted to evaluate the effectiveness and robustness of the proposed RL-DCA, which replaces the traditional signal transformation process in the DCA. The goal is to assess the ability of RL-DCA to dynamically learn and assign signals (PAMP, DS, SS) to features, thereby improving anomaly detection performance across various types of datasets. The experiments were carried out on nine publicly available datasets, which vary in feature dimensionality, data size, and domain

characteristics. Each dataset was pre-processed by removing missing values, encoding categorical features using Label Encoder, and scaling numerical features using

Standard Scaler, following commonly adopted preprocessing practices in anomaly detection studies [38]. To ensure fair comparison and consistency across datasets, the same preprocessing pipeline was applied to all datasets. This approach is widely used in comparative benchmarking studies to avoid dataset dependent bias and to ensure that performance differences are attributed to the learning model rather than preprocessing variations [39].

The experimental protocol involves two main evaluations:

C Reinforcement Learning–Based Signal Transformation

The proposed RL-DCA framework employs a Q-learning agent to dynamically map input features to the DCA signal categories, namely PAMP, DS, and SS. Through interaction with the environment, the agent learns a signal transformation policy that maximizes the expected reward associated with correct anomaly classification. The behaviour of the RL agent is controlled by three main hyperparameters: the learning rate (α), the discount factor (γ), and the exploration rate (ε). These parameters regulate the learning dynamics of the agent, including how new information updates the Q-values, how future rewards are considered, and how the balance between exploration and exploitation is maintained during training.

The learning rate α determines how strongly newly observed rewards influence the update of the Q-values. In this study, $\alpha = 0.3$ was selected to allow gradual updates to the Q-table while avoiding unstable fluctuations during learning. A moderate learning rate helps maintain stable convergence while still allowing the agent to adapt its policy. The discount factor γ controls the contribution of future rewards in the Q-value update. A value of $\gamma = 0.9$ enables the agent to consider long-term rewards while still emphasizing immediate feedback from the environment. This encourages the agent to learn signal categorization strategies that improve overall detection performance over time.

The exploration rate ε defines the probability of selecting a random action instead of the currently optimal action. In this work, $\varepsilon = 0.1$ was used to ensure sufficient exploration of the action space while allowing the agent to increasingly exploit the best-performing signal mappings identified during training.

The selected parameter configuration ($\alpha = 0.3$, $\gamma = 0.9$, $\varepsilon = 0.1$) follows commonly adopted settings in Q-learning-based optimisation methods and was further validated through empirical experimentation. The convergence behaviour of the RL agent is illustrated in Fig. 6, which shows the evolution of the average reward during training on the SP dataset. As training progresses, the learning curve gradually stabilizes, indicating that the RL agent converges toward a consistent signal transformation policy. This behaviour demonstrates that the selected hyperparameter configuration supports stable policy learning within the RL-DCA framework.

D Enhanced DCA Classification

The transformed signals are processed using the standard DCA mechanism, where classification decisions are made based on the Mature Context Antigen Value (MCAV). A decision threshold of 0.85 is applied to distinguish between normal and anomalous antigens. This threshold value has been widely adopted in DCA related studies [40] and was further vali dated experimentally in this work to achieve a suitable balance between detection accuracy and false alarm rate. To ensure statistical reliability, each experiment was repeated for 10 independent runs. Datasets were randomly split into 80% training and 20% testing sets, and 10-fold cross-validation was employed. Model performance was evaluated using multiple classification metrics [41], including accuracy Eq. (14), precision Eq. (15), sensitivity (recall) Eq. (16), specificity Eq. (17), and F1-score Eq. (18), which are standard evaluation measures in anomaly detection and classification research. These metrics are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Sensitivity\ (Recall) = \frac{TP}{TP + FN} \tag{16}$$

$$Specificity = \frac{TN}{TN + FP} \tag{17}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{18}$$

To reduce the effect of randomness due to different data splits and initialization conditions, the reported results represent the mean and standard deviation across the repeated runs. The relatively small standard deviation values observed in the experiments indicate that the proposed RL-DCA model produces stable and consistent performance. Where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. These metrics collectively provide a comprehensive assessment of classification performance by capturing overall correctness, anomaly detection reliability, detection completeness, normal-class recognition capability, and the balance between precision and recall. All simulations were executed using Python 3.10 in PyCharm on a Windows 10 machine with an Intel Core i7-10510U CPU (1.8 GHz) and 16 GB RAM. Visualization of results was performed using the matplotlib library.

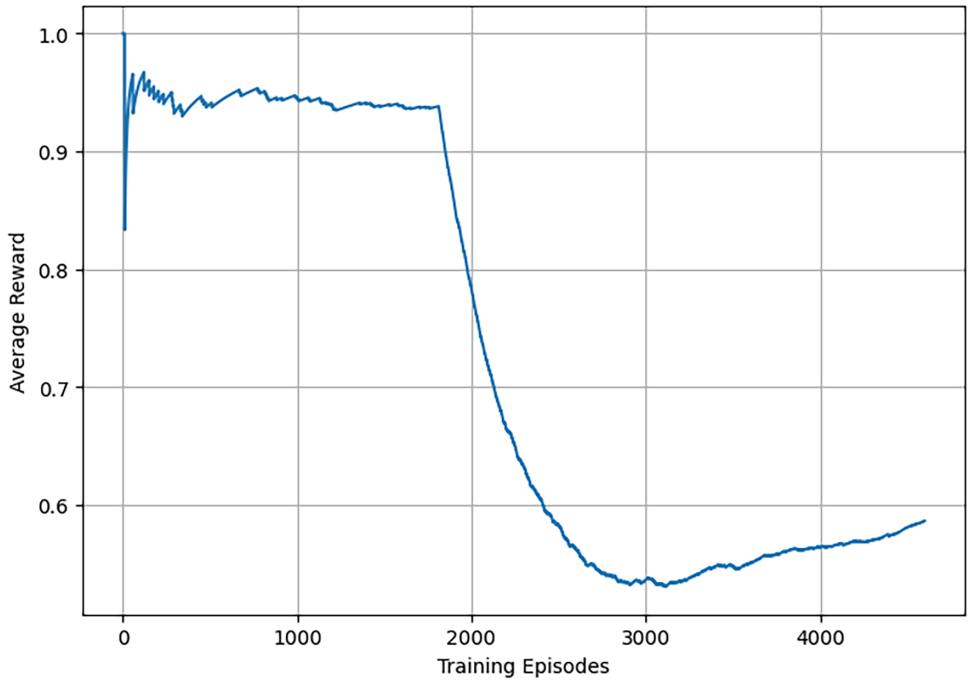

Figure 6: Convergence behaviour of the RL-DCA agent during training.

7 Complexity Analysis

The RL-DCA introduces an RL mechanism for signal categorization prior to executing the standard DCA pipeline. The runtime of RL-DCA therefore depends on the number of iterations over the dataset, the Q-table update operations, and the signal construction steps defined in Algorithm 1. According to Gu et al. [42], the runtime complexity of the standard DCA is bounded by $O(n^2)$, where n is the data size.

Thus, this study focuses on the runtime of the complete RL-DCA. Table 4 shows the details of the primitive operations Algorithm 1, together with the number of times each operation is executed. For each input sample, the algorithm performs state construction, Q-table initialization (if required), ε-greedy action selection, reward calculation, Q-value update, and signal construction. All operations are bound by constant time except the Q-update, which requires computing a maximum over the action space. Since the action space is fixed to three categories PAMP, DS, SS), this operation also executes in constant time. Therefore, the runtime of the Q-learning stage scales linearly with the dataset size n. The runtime of the Q-learning stage can be expressed as Eq. (19):

$$\mathrm{T(n)} = c_0 + \mathrm{n} \times (c_1 + c_2 + c_3 + c_4 + c_5) \tag{19}$$

where c_0 is the initialization cost, and $c_1, \ldots, c_5$ denote the constant costs of state processing, action selection, reward computation, Q-table update, and signal construction. Simplifying, we obtain Eq. (20):

$$T\,(n) = o\,(n) \tag{20}$$

Table 4: Details of primitive operations of Algorithm 1, where n is the dataset size.

Line No.	Description	Times
1	Initialize Q-table	1
2	For-loop over dataset	n
3–5	State formation and Q-table initialization	n
6–9	ε-greedy action selection	n
10	Reward computation	n
11–13	Next-state formation and Q-table initialization	n
14	Q-value update	n
16–21	Construct signal vector (PAMP, Danger, Safe)	n
23	Append signal to output set	n

Since the Q-learning stage is followed by the standard DCA classification phase with complexity $o\left(n^2\right)$, the overall runtime of RL-DCA using is:

$$T_{RL-DCA}\,(n) = o\,(n) + o\left(n^2\right) = o(n^2) \tag{21}$$

As shown in Eq. (21), the worst-case runtime complexity of RL-DCA is $O(n^2)$, dominated by the standard DCA phase. The Q-learning signal mapping contributes only to the linear overhead. To further verify this analysis, the empirical runtime was calculated across multiple benchmark datasets. Fig. 7 illustrates the average execution time of RL-DCA in seconds. The results show that RL-DCA maintains competitive runtime performance, scaling efficiently even on large datasets such as NSL-KDD and UNSW15. Considering the considerable accuracy improvements achieved, the additional computational overhead introduced by the RL mechanism is considered acceptable.

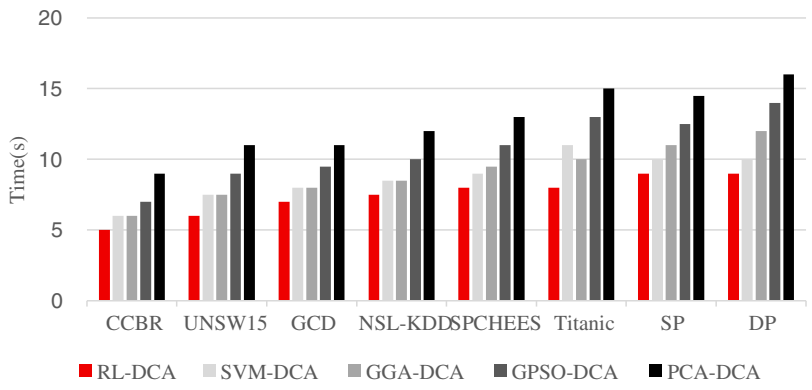

Figure 7: Mean execution time (in seconds) acquired by DCA versions (RL-DCA, GGA-DCA, GPSO-DCA, PCA-DCA, and SVM-DCA) when performing classification across the benchmark datasets.

8 Experimental Results and Discussion

The proposed RL-DCA model was evaluated against six baseline methods, namely GGA-DCA [14], GPSO-DCA [17], PCA-DCA, SVM-DCA, KNN, DT [14] and SVM [43] using nine benchmark datasets spanning biomedical, financial, behavioral, and cybersecurity domains. The evaluation employed multiple performance metrics, including accuracy (Table 5), precision (Table 6), specificity (Table 7), F1-score, and area under the ROC curve (AUC). For all metrics, the reported results represent mean performance values with corresponding standard deviations computed over multiple experimental runs. Tables 5–7 present detailed numerical comparisons of accuracy, precision, and specificity across all datasets. To improve readability and facilitate interpretation of the remaining performance metrics, the F1-score and AUC results are summarized using graphical representations rather than extensive numerical tables.

As shown in Tables 5–7, RL-DCA demonstrates consistently strong performance across mostbench-mark datasets. In terms of classification accuracy, RL-DCA achieves high detection rates on complex datasets such as NSL-KDD, DP, SP, and SPCHEES, outperforming or matching classical DCA variants and conventional machine learning classifiers. Similar trends are observed for precision and specificity, indicating that RL-DCA effectively balances correct positive detection while minimizing false alarms across diverse data characteristics.

Table 5: Mean accuracy (%) ± standard deviation for all algorithms across benchmark datasets.

Dataset	RL-DCA	GGA-DCA	GPSO-DCA	PCA-DCA	SVM-DCA	KNN	DT	SVM
CCBR	86.9 ± 7.3	84.3 ± 1.5	83.2 ± 2.1	81.5 ± 2.4	68.7 ± 4.0	86.2 ± 10.7	83.5 ± 15.0	83.33 ± 6.15
GCD	83.8 ± 2.2	78.6 ± 0.9	79.5 ± 1.7	76.8 ± 1.9	50.8 ± 3.2	58.5 ± 7.1	71.0 ± 2.8	72.55 ± 1.25
Titanic	95.2 ± 1.0	79.1 ± 0.3	81.6 ± 0.7	78.5 ± 0.8	74.9 ± 0.4	74.1 ± 3.9	79.0 ± 1.2	78.73 ± 1.45
DP	97.3 ± 0.7	96.0 ± 0.4	96.5 ± 0.5	95.8 ± 0.6	85.5 ± 1.7	97.6 ± 3.9	96.4 ± 3.9	97.65 ± 1.18
SP	97.3 ± 0.7	94.7 ± 1.5	95.4 ± 0.9	94.1 ± 1.1	88.2 ± 0.7	80.6 ± 3.0	91.6 ± 11.0	85.82 ± 1.12
SPCHEES	96.7 ± 0.8	93.9 ± 1.4	94.5 ± 0.9	93.2 ± 1.0	90.1 ± 1.9	89.5 ± 2.0	88.7 ± 2.1	88.22 ± 1.25
NSL-KDD	99.1 ± 0.5	96.8 ± 0.6	97.1 ± 0.5	95.9 ± 0.7	84.8 ± 2.5	83.6 ± 2.4	82.1 ± 2.6	95.68 ± 0.55
UNSW15	93.9 ± 1.3	92.4 ± 1.0	92.8 ± 0.9	91.5 ± 1.1	86.2 ± 2.2	85.1 ± 2.3	83.5 ± 2.4	95.75 ± 1.08
COIL2000	97.6 ± 0.3	96.2 ± 0.4	96.5 ± 0.3	95.8 ± 0.4	91.7 ± 1.7	90.6 ± 1.6	89.8 ± 1.9	94.05 ± 0.00

Since the reward function in RL-DCA is label-guided during the learning phase, the reinforcement learning component introduces supervised information into the signal categorization process. To provide a fair and comprehensive evaluation, the experimental comparison includes not only traditional DCA variants but also conventional supervised machine learning classifiers such as SVM. This comparison allows the proposed method to be evaluated against both immune-inspired approaches and standard supervised learning baselines.

Table 6: Mean precision (%) ± standard deviation for all algorithms across benchmark datasets.

Dataset	RL-DCA	GGA-DCA	GPSO-DCA	PCA-DCA	SVM-DCA	KNN	DT	SVM
CCBR	73.7 ± 14.7	72.9 ± 4.0	83.9 ± 2.0	82.1 ± 2.3	70.2 ± 3.8	86.7 ± 10.1	84.1 ± 14.7	76.50 ± 17.13
GCD	84.8 ± 1.7	76.6 ± 0.7	80.2 ± 1.6	77.5 ± 1.8	52.1 ± 3.0	59.3 ± 6.8	71.8 ± 2.7	73.64 ± 0.85
Titanic	95.8 ± 0.3	60.1 ± 0.3	82.0 ± 0.6	79.1 ± 0.7	75.3 ± 0.4	74.5 ± 3.7	79.5 ± 1.1	77.29 ± 1.57
DP	99.6 ± 0.5	94.8 ± 0.5	96.7 ± 0.5	96.0 ± 0.6	86.1 ± 1.6	97.9 ± 3.7	96.6 ± 3.7	100.00 ± 0.00
SP	99.6 ± 0.5	89.4 ± 1.3	95.7 ± 0.8	94.5 ± 1.0	88.9 ± 0.6	81.3 ± 2.8	92.0 ± 10.8	87.14 ± 1.44
SPCHEES	97.6 ± 0.6	97.5 ± 0.6	94.7 ± 0.8	93.5 ± 1.0	90.7 ± 1.8	89.9 ± 1.9	89.1 ± 2.0	86.55 ± 1.55
NSL-KDD	99.8 ± 0.2	99.0 ± 0.2	97.3 ± 0.4	96.1 ± 0.6	85.2 ± 2.4	83.9 ± 2.3	82.5 ± 2.5	95.82 ± 0.95
UNSW15	94.6 ± 0.9	94.5 ± 0.9	93.0 ± 0.9	91.8 ± 1.0	86.5 ± 2.1	85.4 ± 2.2	83.8 ± 2.3	95.52 ± 1.23
COIL2000	73.1 ± 2.6	73.5 ± 2.6	96.7 ±0.3	96.0 ± 0.4	91.9 ± 1.6	90.9 ± 1.5	90.1 ± 1.8	0.00 ± 0.00

Table 7: Mean specificity (%) ± standard deviation for all algorithms across benchmark datasets.

Dataset	RL-DCA	GGA-DCA	GPSO-DCA	PCA-DCA	SVM-DCA	KNN	DT	SVM
CCBR	83.1 ± 11.9	82.6± 1.8	84.3 ± 2.0	82.5 ± 2.2	69.8 ± 3.9	87.0 ± 9.9	84.4 ± 14.5	92.73 ± 5.45
GCD	60.6 ± 5.0	79.3 ± 1.0	80.5 ± 1.5	77.9 ± 1.8	51.9 ± 3.1	59.5 ± 6.7	72.0 ± 2.6	20.83 ± 3.82
Titanic	91.3 ± 0.7	69.1 ± 0.4	82.3 ± 0.6	79.3 ± 0.7	75.0 ± 0.4	74.7 ± 3.6	79.7 ± 1.1	39.86 ± 6.29
DP	99.8 ± 0.3	95.6 ± 0.4	96.9 ± 0.4	96.2 ± 0.5	85.9 ± 1.6	98.0 ± 3.6	96.8 ± 3.6	100.00 ± 0.00
SP	99.8 ± 0.3	92.4 ± 1.3	95.8 ± 0.8	94.8 ± 1.0	88.6 ± 0.7	81.5 ± 2.7	92.3 ± 10.6	92.81 ± 0.86
SPCHEES	97.8 ± 0.6	94.4 ± 1.3	94.9 ± 0.8	93.7 ± 0.9	90.5 ± 1.8	90.1 ± 1.9	89.3 ± 1.9	87.31 ± 1.75
NSL-KDD	99.9 ± 0.2	97.2 ± 0.5	97.4 ± 0.4	96.2 ± 0.5	85.0 ± 2.4	84.1 ± 2.3	82.8 ± 2.4	96.73 ± 0.79
UNSW15	58.8 ± 7.1	92.9 ± 0.9	93.2 ± 0.9	92.0 ± 1.0	86.3 ± 2.1	85.6 ± 2.2	84.0 ± 2.2	66.12 ± 9.71
COIL2000	97.8 ± 0.3	96.6 ± 0.3	96.8 ± 0.3	96.1 ± 0.4	91.8 ± 1.6	91.0 ± 1.5	90.3 ± 1.8	100.00 ± 0.00

The comparative F1-score performance of all algorithms across the benchmark datasets is illustrated in Fig. 8. This figure highlights the overall balance between precision and recall achieved by each method. RL-DCA attains superior or competitive F1-scores on most datasets, particularly in cybersecurity and structured domains such as NSL-KDD, SP, DP, and UNSW15. These results demonstrate that the adaptive signal categorization enabled by reinforcement learning enhances the robustness of the DCA framework, especially in complex or high-dimensional environments where static signal mappings and traditional classifiers exhibit reduced effectiveness. RL-DCA demonstrates consistently competitive or superior performance across most datasets, highlighting its ability to balance precision and recall across diverse domains.

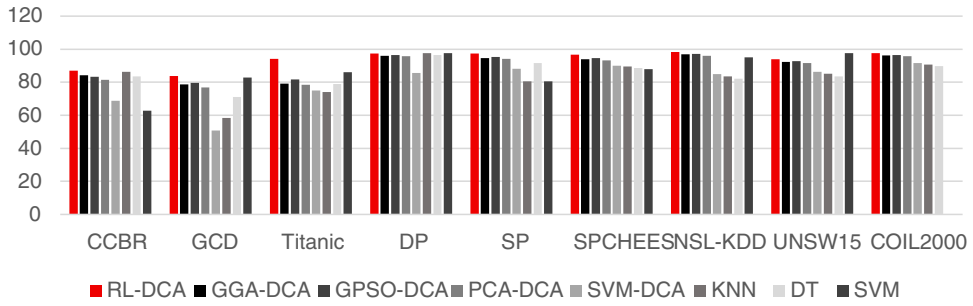

Figure 8: F1-score comparison of all algorithms across benchmark datasets.

The discriminative capability of the evaluated algorithms is further examined using the AUC metric, as shown in Fig. 9. The graphical comparison reveals that RL-DCA maintains strong discrimination performance across most datasets, achieving high AUC values on NSL-KDD, DP, SP, and SPCHEES. Although RL-DCA exhibits relatively lower AUC performance on certain imbalanced financial datasets such as GCD

and COIL2000, it remains competitive with existing DCA variants and conventional classifiers, confirming its ability to generalize across heterogeneous data distributions.

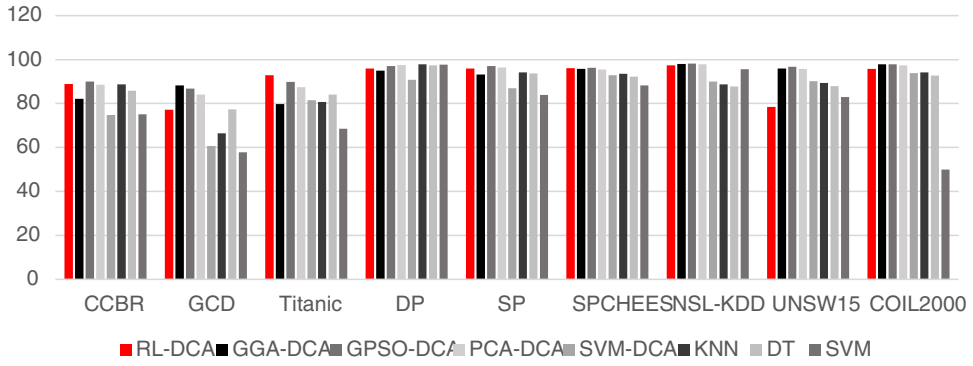

Figure 9: The AUC performance comparison of all algorithms across benchmark datasets.

To further analyze the experimental results, a paired t-test was conducted to determine whether statistically significant differences exist between the proposed RL-DCA model and other signal categorization algorithms of the DCA framework (referred to as "Comparisons"). The statistical test was performed under a significance level of $\alpha = 0.05$.

The following hypotheses were considered:

$$H_0: \mu_{RL-DCA} = \mu_{Comparisons}$$
$$H_1: \mu_{RL-DCA} \neq \mu_{Comparisons}$$

where μ_{RL-DCA} represents the mean classification accuracy obtained by the proposed RL-DCA model, and $\mu_{Comparisons}$ represents the mean accuracy obtained by the baseline algorithms. Table 8 presents the t-test results based on classification accuracy between RL-DCA and other DCA-based signal categorization algorithms. With ten experimental runs, the degree of freedom is df = 9. At a significance level of $\alpha = 0.05$, the corresponding critical t-value from the student's t-distribution table is 2.262. If the calculated t-value is less than 2.262, the null hypothesis H_0 is accepted, indicating that no statistically significant difference exists between the compared methods. Conversely, if the calculated t-value exceeds 2.262, the null hypothesis is rejected, indicating a statistically significant performance difference. As shown in Table 8, all calculated t-values are greater than 2.262, demonstrating that the proposed RL-DCA model achieves statistically significant improvements compared with the other signal categorization algorithms of the DCA framework across the evaluated benchmark datasets. These results confirm that the reinforcement learning based signal categorization mechanism effectively enhances the classification capability of the traditional DCA model.

9 Threshold Sensitivity Analysis

In the traditional DCA, classification is performed using the MCAV, where samples are labeled as anomalous when their MCAV exceeds a predefined threshold θ. Previous studies commonly adopted a threshold value of $\theta = 0.85$, however, the influence of this parameter on detection performance requires further investigation. Therefore, a threshold sensitivity analysis was conducted by varying the MCAV threshold within the range:

$$\theta \in \{0.50, 0.65, 0.75, 0.85, 0.95\}$$

Table 8: Details of primitive operations of Algorithm 1, where n is the dataset size.

Category	Dataset	RL-DCA			
		GGA-DCA	GPSO-DCA	PCA-DCA	SVM-DCA
1	CCBR	6.51	7.00	7.07	8.74
2	GCD	30.25	34.91	38.24	25.78
3	Titanic	26.26	48.16	40.11	27.76
4	DP	11.23	8.85	8.97	15.84
5	SP	44.07	107.00	48.17	27.96
6	SPCHEES	29.43	53.18	29.56	38.63
7	NSL-KDD	204.52	234.75	116.55	182.22
8	UNSW15	36.87	158.68	127.69	134.57
9	COIL2000	25.24	18.24	21.62	27.88

The analysis was performed on two representative datasets with different class distributions. The NSL-KDD dataset, which contains relatively balanced attack and normal samples, was selected to represent balanced scenarios. In contrast, the COIL2000 dataset, which is highly imbalanced, was used to evaluate the behavior of the algorithm under skewed class distributions.

Fig. 10 illustrates the F1-score obtained under different threshold values for both datasets. The results indicate that the proposed RL-DCA model maintains relatively stable performance across a wide range of threshold settings. For the NSL-KDD dataset, the performance remains consistently high across all evaluated thresholds, indicating strong separability between normal and attack traffic. For the COIL2000 dataset, the F1-score improves as the threshold increases from 0.50 to approximately 0.65 and remains relatively stable between 0.65 and 0.95.

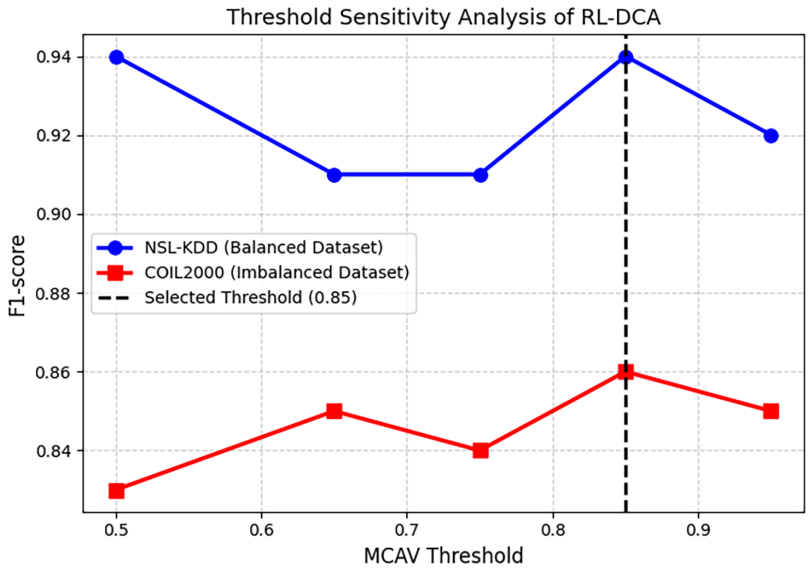

Figure 10: The F1-score obtained under different threshold values for both datasets.

Overall, the results demonstrate that RL-DCA is not overly sensitive to moderate variations in the MCAV threshold. Among the tested values, the threshold 0.85 achieves the highest or near-highest F1-score

across the evaluated datasets, providing a balanced trade-off between anomaly detection capability and false positive control. Therefore, this threshold was selected for the subsequent experiments.

Furthermore, the threshold sensitivity analysis highlights a limitation of the traditional DCA classification mechanism: the final decision depends on a manually selected threshold parameter, which may not adapt optimally to different data distributions.

10 Discussion

This section discusses the experimental results of the proposed RL-DCA model and explains their implications with respect to the research question. Instead of repeating the numerical values reported in Tables 5–7, the discussion focuses on the main performance patterns, the observed trade-offs, and the limitations identified across the evaluated datasets.

Overall, the results show that RL-DCA achieves the highest or near-highest performance on most datasets and evaluation metrics. The improvements are particularly clear in datasets with high dimensionality or complex feature relationships, such as NSL-KDD and UNSW15. In these datasets, traditional DCA variants that rely on static signal mappings or heuristic optimization methods tend to perform less effectively. The results suggest that learning the signal categorization directly from classification feedback helps the model generate more informative signals. As a result, RL-DCA improves the ability of the DCA framework to distinguish between normal and anomalous patterns across different application domains.

On structured datasets such as DP, RL-DCA achieves performance comparable to the best classical machine learning classifiers while showing lower variation across repeated runs. This indicates that the reinforcement learning component converges to a stable signal mapping strategy and reduces sensitivity to initialization. In contrast, conventional classifiers tend to show larger performance degradation on cybersecurity datasets. This behaviour is consistent with known challenges in these datasets, such as class imbalance, sparse features, and complex attack patterns.

Although RL-DCA improves performance in most cases, the results also reveal several dataset-dependent limitations. For example, the GCD and COIL2000 datasets show relatively high accuracy but lower F1-score and precision. This behaviour is mainly caused by class imbalance, where the large number of majority-class instances makes it more difficult to correctly identify rare events. A similar trade-off appears in the UNSW15 dataset. In this case, RL-DCA achieves strong accuracy, but slightly lower specificity and AUC compared with GGA-DCA and GPSO-DCA. This suggests that the model tends to prioritize detecting anomalous instances, which increases sensitivity but may also produce more false positives. Such behaviour can be useful in security applications where missing an attack is more critical than raising additional alarms. However, it also highlights the need for better balance between sensitivity and specificity when dealing with highly imbalanced datasets.

The results provide a clear answer to the research question of this study. The findings show that replacing fixed signal mappings with a learning-based signal categorization mechanism allows the DCA preprocessing stage to become more adaptive and less dependent on manual configuration. By learning signal assignments directly from classification outcomes, RL-DCA can adjust to different dataset characteristics and maintain competitive performance across multiple domains.

It should be noted that the reinforcement learning component uses class labels within the reward function to guide the learning of signal assignments. Therefore, the signal categorization stage becomes label guided during training. However, reinforcement learning is used only to optimize the preprocessing stage of the DCA, while the final anomaly decision is still performed by the original DCA context assessment and MCAV classification mechanism.

The findings also support the main contributions of this work. First, the proposed adaptive signal categorization mechanism improves anomaly detection performance in most evaluated datasets. Second, the experiments conducted on nine benchmark datasets demonstrate that the approach can generalize across different application areas. Third, the runtime analysis shows that the additional computational cost introduced by the learning mechanism remains moderate and acceptable compared with the benefits of removing manual signal design.

Despite these advantages, several limitations remain. The results show that the model is still affected by class imbalance in some datasets, particularly GCD, COIL2000, and UNSW15, where minority-class detection performance is reflected by lower F1-score, specificity, or AUC values. In addition, the current classification stage still relies on a fixed MCAV threshold. A single threshold value may not always be optimal for datasets with different class distributions. Future work will therefore focus on improving the classification stage of the DCA and exploring adaptive threshold strategies. A data-driven threshold adjustment mechanism could further improve the balance between detection sensitivity and false-positive rates.

11 Conclusion

This study proposes a new adaptive framework, called RL-DCA, which integrates reinforcement learning with the DCA to achieve dynamic signal categorization. Unlike traditional DCA approaches that rely on fixed rules, static feature transformations, or manually designed preprocessing, the proposed method learns the mapping between features and immunological signals directly from data. By using Q-learning, the RL agent iteratively updates signal assignments based on classification feedback, allowing the model to automatically discover more informative signal representations for anomaly detection.

The proposed framework preserves the immune-inspired structure of the original DCA while improving its preprocessing stage through an adaptive learning mechanism. In RL-DCA, the agent explores different signal assignments for PAMP, DS, and SS using a Q-table and an ε-greedy learning strategy. This dynamic process allows the model to adjust signal categorization according to the characteristics of the dataset, reducing dependence on manual configuration and improving generalization.

Experimental results on nine benchmark datasets from healthcare, finance, social behavior, spam detection, and cybersecurity domains demonstrate the effectiveness of the proposed approach. RL-DCA consistently achieves higher or competitive performance compared with existing DCA variants, including PCA-DCA, GGA-DCA, and GPSO-DCA, as well as classical machine learning classifiers such as SVM, KNN, and Decision Tree. These results show that adaptive signal categorization improves anomaly detection performance and enhances the robustness of the DCA framework across different application domains.

Future work will focus on extending the RL-DCA framework to support multi-class classification and online anomaly detection in streaming environments. Such extensions will enable the model to operate in real-time scenarios and further improve its applicability to practical problems such as network intrusion detection, fraud detection, and adaptive cybersecurity monitoring.

Overall, the results demonstrate that incorporating reinforcement learning into the DCA provides an effective way to transform static signal categorization into a data-driven and adaptive process, improving the flexibility and generalization capability of immune-inspired anomaly detection systems.

Acknowledgement: The authors would like to express their appreciation to the Center for Artificial Intelligence Technology (CAIT), Universiti Kebangsaan Malaysia (UKM) for providing the research environment and facilities that supported this study.

Funding Statement: This research was supported by the Fundamental Research Grant Scheme (FRGS) under the Ministry of Higher Education Malaysia, grant number FRGS/1/2023/ICT02/UKM/02/2.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design were carried out by Yousra Abudaqqa, Zulaiha Ali Othman, and Azuraliza Abu Bakar. Yousra Abudaqqa conducted the data collection, implementation, and experimental analysis. Interpretation and validation of the results were performed by Yousra Abudaqqa, Zulaiha Ali Othman, and Azuraliza Abu Bakar. Yousra Abudaqqa prepared the initial draft of the manuscript. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this study are publicly available benchmark datasets widely used in machine learning and anomaly detection. These datasets are available from publicly accessible repositories through the UCI Machine Learning Repository and Kaggle. The datasets are available at: UCI Machine Learning Repository: https://archive.ics.uci.edu/, Kaggle Datasets: https://www.kaggle.com/datasets.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Myakala PK, Bura C, Jonnalagadda AK. Artificial immune systems: a bio-inspired paradigm for computational intelligence. J Artif Intell Big Data. 2025;5(1):1–13. doi:10.31586/jaibd.2025.1233.
2. Widuliński P. Artificial immune systems in local and network cybersecurity: an overview of intrusion detection strategies. Appl Cybersecur Internet Gov. 2023;2(1):184306. doi:10.60097/acig/162896.
3. Chelly Z, Elouedi Z. A survey of the dendritic cell algorithm. Knowl Inf Syst. 2016;48(3):505–35. doi:10.1007/s10115-015-0891-y.
4. Greensmith J, Aickelin U, Cayzer S. Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In: Artificial immune systems. Berlin/Heidelberg, Germany: Springer; 2005. p. 153–67. doi:10.1007/11536444_12.
5. Belhadj M, Cherif F, Cheriet M. NMF-DCA: an efficient dendritic cell algorithm based on non-negative matrix factorization. Int J Comput Digit Syst. 2021;10:575–83.
6. Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: a survey. J Artif Intell Res. 1996;4:237–85. doi:10.1613/jair.301.
7. Clifton J, Laber E. Q-learning: theory and applications. Annu Rev Stat Appl. 2020;7(1):279–301. doi:10.1146/annurev-statistics-031219-041220.
8. Mohsin MFM, Hamdan AR, Abu Bakar A. An upper and lower CUSUM for signal normalization in the dendritic cell algorithm. Evol Intell. 2016;9(1):37–51. doi:10.1007/s12065-016-0136-3.
9. Farzadnia E, Shirazi H, Nowroozi A. A new intrusion detection system using the improved dendritic cell algorithm. Comput J. 2021;64(8):1193–214. doi:10.1093/comjnl/bxaa140.
10. Greensmith J, Aickelin U, Tedesco G. Information fusion for anomaly detection with the dendritic cell algorithm. Inf Fusion. 2010;11(1):21–34. doi:10.1016/j.inffus.2009.04.006.
11. Al-Hasan AA, El-Alfy EM. Dendritic cell algorithm for mobile phone spam filtering. Procedia Comput Sci. 2015;52(3):244–51. doi:10.1016/j.procs.2015.05.067.
12. Pereira VE. Automatic signal characterization for the Dendritic Cell Algorithm [dissertation]. Porto, Portugal: Universidade do Porto; 2024.
13. Chelly Z, Elouedi Z. Improving the dendritic cell algorithm performance using fuzzy-rough set theory as a pattern discovery technique. In: Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014. Cham, Switzerland: Springer International Publishing; 2014. p. 23–32. doi:10.1007/978-3-319-08156-4_3.
14. Zhang D, Liang Y, Dong H. Dendritic cell algorithm with grouping genetic algorithm for input signal generation. Comput Model Eng Sci. 2023;135(3):2025–45. doi:10.32604/cmes.2023.022864.

15. Elisa N, Yang L, Chao F, Naik N. A comparative study of genetic algorithm and particle swarm optimisation for dendritic cell algorithm. In: 2020 IEEE Congress on Evolutionary Computation (CEC); 2020 Jul 19–24; Glasgow, UK. p. 1–8. doi:10.1109/cec48606.2020.9185497.
16. Elisa N, Yang L, Naik N. Dendritic cell algorithm with optimised parameters using genetic algorithm. In: 2018 IEEE Congress on Evolutionary Computation (CEC); 2018 Jul 8–13; Rio de Janeiro, Brazil. p. 1–8. doi:10.1109/CEC.2018.8477932.
17. Zhang D, Liang Y. Dendritic cell algorithm with group particle swarm optimization for input signal generation. In: PRICAI 2021: trends in artificial intelligence. Cham, Switzerland: Springer International Publishing; 2021. p. 527–39. doi:10.1007/978-3-030-89188-6_39.
18. Zhang D, Zhang Y, Liang Y. Dendritic cell algorithm with Bayesian optimization hyperband for signal fusion. Comput Mater Contin. 2023;76(2):2317–36. doi:10.32604/cmc.2023.038026.
19. Zhou W, Liang Y. An immune optimization based deterministic dendritic cell algorithm. Appl Intell. 2022;52(2):1461–76. doi:10.1007/s10489-020-02098-0.
20. Chen G, Shuo P, Rong J, Chao L. An anomaly detection system based on dendritic cell algorithm. In: 2009 Third International Conference on Genetic and Evolutionary Computing; 2019 Oct 14–17; Guilin, China. p. 192–5. doi:10.1109/WGEC.2009.129.
21. Abudaqqa Y, Ali Othman Z, Abu Bakar A. Enhancing dendritic cell algorithm by integration with multi-layer perceptron for anomaly detection. Int J Adv Comput Sci Appl. 2025;16(7):1035. doi:10.14569/ijacsa.2025.0160798.
22. Yee Por L, Dai Z, Juan Leem S, Chen Y, Yang J, Binbeshr F, et al. A systematic literature review on AI-based methods and challenges in detecting zero-day attacks. IEEE Access. 2024;12:144150–63. doi:10.1109/access.2024.3455410.
23. Dai Z, Por LY, Chen YL, Yang J, Ku CS, Alizadehsani R, et al. An intrusion detection model to detect zero-day attacks in unseen data using machine learning. PLoS One. 2024;19(9):e0308469. doi:10.1371/journal.pone.0308469.
24. Liu DR, Li HL, Wang D. Feature selection and feature learning for high-dimensional batch reinforcement learning: a survey. Int J Autom Comput. 2015;12(3):229–42. doi:10.1007/s11633-015-0893-y.
25. Ren K, Zeng Y, Cao Z, Zhang Y. ID-RDRL: a deep reinforcement learning-based feature selection intrusion detection model. Sci Rep. 2022;12(1):15370. doi:10.1038/s41598-022-19366-3.
26. Yang X, Howley E, Schukat M. Agent-based dynamic thresholding for adaptive anomaly detection using reinforcement learning. Neural Comput Appl. 2025;37(23):18775–91. doi:10.1007/s00521-024-10536-0.
27. Basak H, Das M, Modak S. RSO: a novel reinforced swarm optimization algorithm for feature selection. In: IEEE EUROCON 2021 19th International Conference on Smart Technologies; 2021 Jul 6–8; Lviv, Ukraine. p. 203–8. doi:10.1109/eurocon52738.2021.9535639.
28. Roslan SNRB, Huddin AB, Bahari MRK, Zaki WMDW, Mokri SS. Reinforcement learning for automated anterior commissure localization in brain CT scans for tumor detection. In: 2025 21st IEEE International Colloquium on Signal Processing & Its Applications (CSPA); 2025 Feb 7–8; Pulau Pinang, Malaysia. p. 123–8. doi:10.1109/CSPA64953.2025.10933268.
29. Ng DV, Hwang JG. Android malware detection using the dendritic cell algorithm. In: 2014 International Conference on Machine Learning and Cybernetics; 2014 Jul 13–16; Lanzhou, China. p. 257–62. doi:10.1109/ICMLC.2014.7009126.
30. Gu F. Theoretical and empirical extensions of the dendritic cell algorithm [dissertation]. Nottingham, UK: University of Nottingham; 2011.
31. Chelly Z, Elouedi Z. RC-DCA: a new feature selection and signal categorization technique for the dendritic cell algorithm based on rough set theory. In: Artificial immune systems. Berlin/Heidelberg, Germany: Springer; 2012. p. 152–65. doi:10.1007/978-3-642-33757-4_12.
32. Xie H, Li J, Zhang Q, Wang Y. Comparison among dimensionality reduction techniques based on Random Projection for cancer classification. Comput Biol Chem. 2016;65(18):165–72. doi:10.1016/j.compbiolchem.2016.09.010.
33. Boucherie RJ, van Dijk NM. Markov decision processes in practice. Cham, Switzerland: Springer International Publishing; 2017. doi:10.1007/978-3-319-47766-4.

34. Frank A, Asuncion A. UCI machine learning repository [Internet]. [cited 2026 Jan 1]. Available from: https://archive.ics.uci.edu/.
35. Triguero I, González S, Moyano JM, García S, Alcalá-Fdez J, Luengo J, et al. KEEL 3.0: an open source software for multi-stage analysis in data mining. Int J Comput Intell Syst. 2017;10(1):1238. doi:10.2991/ijcis.10.1.82.
36. Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS); 2015 Nov 10–12; Canberra, Australia. p. 1–6. doi:10.1109/MilCIS.2015.7348942.
37. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications; 2009 Jul 8–10; Ottawa, ON, Canada. p. 1–6. doi:10.1109/CISDA.2009.5356528.
38. Davis JJ, Clark AJ. Data preprocessing for anomaly based network intrusion detection: a review. Comput Secur. 2011;30(6–7):353–75. doi:10.1016/j.cose.2011.05.008.
39. Lu J. The elements of statistical learning: data mining, inference, and prediction. Oxford, UK: Oxford University Press (OUP); 2010.
40. Elisa N, Yang L, Fu X, Naik N. Dendritic cell algorithm enhancement using fuzzy inference system for network intrusion detection. In: 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE); 2019 Jun 23–26; New Orleans, LA, USA. p. 1–6. doi:10.1109/fuzz-ieee.2019.8859006.
41. Chicco D, Jurman G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics. 2020;21(1):6. doi:10.1186/s12864-019-6413-7.
42. Gu F, Greensmith J, Aickelin U. Theoretical formulation and analysis of the deterministic dendritic cell algorithm. Biosystems. 2013;111(2):127–35. doi:10.1016/j.biosystems.2013.01.001.
43. Chandra MA, Bedi SS. Survey on SVM and their application in imageclassification. Int J Inf Technol. 2021;13(5):1–11. doi:10.1007/s41870-017-0080-1.