Negative Selection Algorithm for Anomaly Detection

Dipankar Dasgupta

Professor, Department of Computer Science
Director, Intelligent Security Systems Research Lab.
University of Memphis
Role of Biological Immune System (BIS)

- Its primary role is to distinguish the host (body cells) from external entities (pathogens).
- When an entity is recognized as non-self (or dangerous) - activates several defense mechanisms leading to its destruction (or neutralization).
- Subsequent exposure to similar entity results in rapid immune response (Secondary Response).
- Overall behavior of the immune system is an emergent property of many local interactions.
An abstract view of BIS:
Multi-Level Detection

Skin & Mucous Membranes

Innate Immunity

Adaptive Immunity
From the computational point of view, the immune system is a

- Distributed information processing system
- Novel pattern recognizer: Self/non-self (Danger) Discrimination
- Multi-level Self regulated Defense System
- Having unique mechanisms for
  - Decentralized control
  - Signaling and Message-passing
  - Co-stimulation
  - Learning and memory
  - Diversity
Computational Models & Algorithms

- **Immune Network Models** (Jerne’74)
- **Negative Selection Algorithms** (Forrest’94)
- **Immune Gene Libraries** (Hightower’90)
- **Associative Memory** (Gilbert’94, Smith’96)
- **Artificial Immune Systems** (Hunt’95, Timmis’97)
- **Immune Agent Architecture** (Mori’98, Dasgupta’99)
- **Artificial Germinal Centers** (Dasgupta’02)
- **Other Models** (Farmer’86, Bersini’90, Varela,’91, etc.)
# Artificial Immune Systems (AIS)

<table>
<thead>
<tr>
<th>Immunological Aspect</th>
<th>Computational Problem</th>
<th>Typical Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self/non-self recognition (NSA)</td>
<td>Anomaly or change detection</td>
<td>- Computer security</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Fault detection</td>
</tr>
<tr>
<td>Immune Networks (AINE, RLAIS, AIRS, FuzzyAIS)</td>
<td>Learning (supervised and unsupervised)</td>
<td>- Classification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Clustering</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Data analysis</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Stream data-mining</td>
</tr>
<tr>
<td>Clonal selection (Clonalg, aiNet)</td>
<td>Search, optimization</td>
<td>- Function optimization</td>
</tr>
<tr>
<td>Cell Mobility (ImmAg)</td>
<td>Distributed processing</td>
<td>- SecAgent architectures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Decentralized robot control</td>
</tr>
</tbody>
</table>
Negative Selection Algorithm (NSA)  
(Forrest ‘94)

An algorithm for change detection based on the principles of self-nonself discrimination (by T Cell receptors) in the immune system. The receptors can detect antigens.

Partition of the Universe of Antigens

SNS: self and nonself \((a \text{ and } b)\)

<table>
<thead>
<tr>
<th>Responses to each set predicted by:</th>
<th>SNS</th>
<th>INS</th>
<th>Danger</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
</tbody>
</table>
Illustration of NS Algorithm:

For binary representation:

- There exists efficient BNS algorithm that runs on linear time with the size of self (D’haeseleer’96).
  - Efficient algorithm to count number of holes.
  - Theoretical analysis based on Information Theory.

D. Dasgupta
Defining the Negative Selection Algorithm (NSA):

- Define \textit{Self} as a normal pattern of activity or stable behavior of a system/process
  - A collection of logically split segments (equal-size) of pattern sequence.
  - Represent the collection as a multiset \( S \) of strings of length \( l \) over a finite alphabet.
- Generate a set \( R \) of \textit{detectors}, each of which fails to match any string in \( S \).
- Monitor new observations (of \( S \)) for changes by continually testing the detectors matching against representatives of \( S \). If any detector ever matches, a change (or deviation) must have occurred in system behavior.
NS Greedy Algorithm: (D’haeseleer’96)

It can generate a diverse set of detectors to provide better coverage in the non-self space. Particularly, instead of generating detectors randomly (in the second phase), the greedy algorithm chooses detectors that are far apart, in order to avoid possible overlapping of detectors and to provide enough coverage in the non-self space.
Partial Matching Rule  
\((r\text{-contiguous symbols})\)

\[
\begin{align*}
X: & \quad \text{ABCDEFGHDEABCE} \\
Y: & \quad \text{BCGCBCACAEABAE}
\end{align*}
\]

Choose a threshold \((r)\):

\[
\text{match} (X, Y) = T \quad r \leq 3
\]

\[
P_M = m^{-r} \left[ (l - r) (m-1) / m + 1 \right]
\]

\(m = \text{size of alphabet}\)

\(l = \text{num of symbols in string}\)

\[
e.g.: \text{strings of length } l=30, \text{matching length } r=8
\]

\[
010101001001110001111110100
\]

\[
111010101101110010100010011110
\]
Anomaly Detection in Time Series

- Dasgupta & Forrest (1996) on time series data, based on the previously discussed *negative-selection algorithm*.

1. collect sufficient time series data to exhibit the normal behavior of a system;
2. determine the range of variation of data and perform a binary encoding according to the desired precision;
3. select a suitable window (concatenation of a fixed number of data points) size which captures the regularities of interest;
4. slide the window along the time series, in non-overlapping steps, and store the encoded string for each window as *self*, from which detectors will be generated;
5. generate a set of detectors that do not match any of the self strings;
6. once a unique set of detectors is generated from the normal database of patterns, it can probabilistically detect any change (or abnormality) in patterns of unseen data;
7. while monitoring the system, use the same encoding scheme for the new data patterns. If a detector is activated, a change in behavior has occurred and an alarm might signal.
Anomaly Detection Process

real data

symbolic representation (binary or other alphabet)

slide window (of size $l$, shift $k$)

symbolic self patterns
local normal behavior of the system
Analyzing the Expressiveness of Binary Matching Rules

- 2-dimensional Euclidean problem space
- NS with binary rules is applied
- The generated detectors are mapped back to the problem space
- Self set: a section of Mackey-Glass data set
Problem Space Representation

Problem Space

Self/non-self Space

(0.4, 0.6) → (102, 53) transform to integer

Binary or Gray encoding

00000000 00000000

01100110 10011001

11111111 11111111
Generated Coverings

\[ \begin{array}{c}
\text{Binary} \\
\text{Gray}
\end{array} \]

\[ \begin{array}{c}
r-\text{contiguous} \\
r-\text{chunk}
\end{array} \]

\[ \begin{array}{c}
10010100 \\
11010110
\end{array} \]

\[ \begin{array}{c}
10010100 \\
**0101**
\end{array} \]

\[ \begin{array}{c}
1 0 0 1 0 1 0 0 \\
1 1 0 1 0 1 1 0 \\
1+0+1+1+1+1+0+1=6
\end{array} \]

\[ \begin{array}{c}
10101000 \\
11010110
\end{array} \]

\[ \begin{array}{c}
1 0 1 0 1 0 0 \\
1 1 0 1 0 1 1 0
\end{array} \]

\[ 1+0+1+1+1+0+1=6 \]
Shape of Binary Matching Rules

- **r-contiguous**
- **r-chunk**
- **Hamming**

**Binary**

- $r = 4$
- $r = 8$

**Gray**

- $r = 4$
- $r = 8$

- $11000001010100011$
- $100000010000000$
- $100100010001000$
- $****0001000****$
- $1100001010100011$
- $100000010000000$
- $11111111111 = 11$
Coverings Generated by Different Values of $r$

$r = 6$

$r = 7$

$r = 8$

$r = 9$
Limitations of BMRs in NSA

- Binary matching rules are not able to capture the semantics of some complex self/non-self spaces.
- It is not easy to extract meaningful domain knowledge.
- Scalability issues: In some cases, large number of detectors are needed to guarantee a good level of detection.
- It is difficult to integrate the NS algorithm with other immune algorithms.
- Crisp boundary of self and non-self may be hard to define
Advances in NSA:

Developments in NSA

- New representation
- New detector generation algorithms
- Non-crisp self/non-self distinction

Hybrid Immune Learning Algorithm

Hyper-rectangles
- Crisp If-Then rules

Fuzzy If-Then rules

Hyper-spheres

NSDR:
- Seq Niching
- Det. Crowding

NSFDR

RNS

RRNS

Multi-shaped
Real-Valued Self/Non-self Space

Use of a multi-dimensional real representation of the space:

– Appropriate for diverse applications
– Some geometrical properties of $\mathbb{R}^n$ that may speed up the negative selection
– It is easier to map the detectors back to the problem space
– Other AIS approaches use this kind of representation
Evolving Fault detectors

• Goal: to evolve 'good' fault indicators (detectors) in the non-self (abnormal) space.

• 'good' detector means:
  – It must not cover self.
  – It has to be as general as possible: the larger the volume, the better.
  – Collectively provide maximum coverage of the non-self space with minimum overlap

• Some detectors serve as specialized (signature for known fault conditions) and others are for probable (or possible) faulty conditions.
RNS Algorithm: Flow Diagram

Begin

For each detector 'd'

Does 'd' match any self point?

Yes

'd.iter' > 't' ?

Yes

Discard 'd'

No

'd.iter'++

Move 'd' away from self

No

Move 'd' away from other detectors

'd.iter' = 0

End

D. Dasgupta
NS Rule Evolution: Different Levels of Deviation

- Define different levels of variability on the self set.
- Evolve detectors for the different levels.

D. Dasgupta
A Heuristic Algorithm for Generating Hyper-spherical Detectors (RNS)

Self Data

Generate random population of detectors

Optimize detector distribution
Generation of Detector using Genetic Algorithm

1. Generate Initial population
2. Choose two parents and cross them
3. Replace closest parent if fitness is better.
Multi-shaped detectors
Anomaly Detection Function

\[ \mu_{\text{self}} : \mathbb{R}^n \rightarrow \text{Range} \]

- Crisp
- Non-crisp discrete
- Fuzzy
Immunity Based Fault Detection

Pilot Input

Reference Model

NASA IFC

Sensors

Monitored System

pre-processing (Data Fusion)

Generate Self

LEARNING

Generate detectors

Detection

New Samples

Feature

Normal data

F1

F2

F3

F4

Self

Alarm

Concept Illustration
Immune-Based Fault Detection System

MILD v1.0

Detector Generator
Fault Detection
EXIT
<table>
<thead>
<tr>
<th>Fault Type</th>
<th>Activated Detectors</th>
<th>Detection Rate (mean)</th>
<th>False Alarm (mean)</th>
<th>False Alarm (std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Engine</td>
<td>10</td>
<td>97.8 %</td>
<td>0.15 %</td>
<td>0.33%</td>
</tr>
<tr>
<td>Tail 3</td>
<td>9</td>
<td>94.7 %</td>
<td>0.76 %</td>
<td>0.26%</td>
</tr>
<tr>
<td>Tail 1</td>
<td>7</td>
<td>91.8 %</td>
<td>1.04%</td>
<td>0.47%</td>
</tr>
<tr>
<td>Wing 3</td>
<td>3</td>
<td>95.6 %</td>
<td>0.43%</td>
<td>0.36%</td>
</tr>
</tbody>
</table>
Testing of two different faults (Tail and wing failure)

<table>
<thead>
<tr>
<th>Type of Fault</th>
<th>Tail</th>
<th>Wing</th>
</tr>
</thead>
<tbody>
<tr>
<td># of activated detectors</td>
<td>82</td>
<td>108</td>
</tr>
<tr>
<td>Detection rate (mean)</td>
<td>89%</td>
<td>92%</td>
</tr>
<tr>
<td>Detection rate (Std)</td>
<td>1.43</td>
<td>1.67</td>
</tr>
<tr>
<td>False Alarm (mean)</td>
<td>0.87%</td>
<td>0.98%</td>
</tr>
<tr>
<td>False Alarm (Std)</td>
<td>0.45</td>
<td>0.32</td>
</tr>
</tbody>
</table>
Variable size Fault Detectors

Radius of 368 detectors

Number of Detectors

Detector Radius (range)

Number of specialized detectors activation

Engine, Tail_1, Tail_2, Wing_2, Wing_1
Combining Negative Selection (NS) and Classification Techniques for Anomaly Detection (Gonzalez’02)

• The idea is to combine conventional classification algorithms and Artificial Immune Systems techniques to perform anomaly detection.
  – In many anomaly detection applications, only positive (normal) samples are available at the training stage.
  – Conventional classification algorithms need positive and negative samples.
  – The proposed approach uses the positive (normal) samples to generate negative samples that are used as training data for a neural network.
Generating Classifier dataset

Diagram:

1. Normal Samples
2. Negative Selection Algorithm
3. Negative Samples
4. Classification Algorithm
5. Training
6. New Samples
7. Anomaly Detection Function
8. Detection
9. Normal
10. Abnormal
Advantages of Negative Selection

• From an information theory point of view, characterization of the normal space is equivalent to the characterization of the abnormal space.

• Distributed detection: Different set of detectors can be distributed at different location

• Other possibilities
  – Generalized and specialized detectors
  – Dynamic detector sets
  – Detectors with specific features
  – Artificial Fault signatures
  – Data samples for classification techniques
Multilevel Immune Learning Algorithm
D. Dasgupta 40
MILA Algorithm
Overview

Initialization
- Collect normal behavior of a system and define self
- Generate different types of detectors from the self
  - $T_d$-detector (specificity)
  - $T_{ar}$-detector (bit level)
  - B-detector (Surface feature)
  - APC-detector (high level)

Recognition
- Sample new data pattern and define antigens
- Check with different type of detectors
  - Error signal of a system captured by APC
  - No suppression from all $T_d$ detectors occurs
  - Any $T_{ar}$-detector activates (match)
  - Any B-detector activates (match)

Logic Operator ($\land, \lor$)
- Antigens are recognized, action follows

Evolutionary
- Cloning, Mutation and Selection

Response
MILA Algorithm Implementation: Basic Strategies

- **Shape-space model:**
  e.g., Ag or Ab is represented as $m = < m_1, m_2, ..., m_L >$

- **Euclidean distance:**
  calculate the degree of Ag-Ab interaction.

- **Partial matching rule:**

D. Dasgupta
Algorithm Implementation: Basic Strategies

- APC recognition: default
- $T_h$ recognition: low-level
- $T_s$ recognition: suppression
- B recognition: high-level
- Cloning and mutation
  - targeted (not blind) cloning
  - positive selection (higher affinity) and negative selection (self tolerant)
Low Level $T_h$ recognition

Peptide length $= k = 4$
High Level B recognition

Self-pattern

Processing: <a1, a2, a3, ... aL>

Self-feature

Matching

<1, a1>, (3, a3), (L, aL)>

<2, a2>, (4, a4), (9, a9)>

<..., ..., ... >

No

Candidate B

Processing: <(1, b1), (3, b3), (L, bL)>

<2, b2>, (4, b4), (9, b9)>

<..., ..., ... >

No

Matured B detector

<1, m1>, (3, m2), (L, mL)>

<..., ..., ... >

Protein Surface

B Cell Receptor
Mysterious Cell---- Ts cell

- Ts exactly exists in body and suppresses immune response! Ts has specificity for special antigen.

- Mechanism remains unknown

- For the problem of anomaly detection, $T_s$ detector is regarded as a special self-detecting agent.

  ✓ *Initialization* phase: $T_s$ detector will be selected if it still matches the self-antigen under more stringent threshold.

  ✓ *Recognition* phase: the response will be terminated when $T_s$ detector matches a special antigen resembling self-data pattern.
Dynamic detector sets

Normal Sample

Testing Sample

1

ROC 1

2

ROC 2

3

ROC 3

4

ROC 4

5

ROC 5

Dynamic Detector set
New Features of MILA

- Combines several immunological metaphors instead of implementing in a piecemeal manner.
- Uses multiple strategies for detection to make the system either very sensitive to any changes or robust to noise.
- Detector generation is problem dependent: different threshold parameters are available tuning the system performance.
New Features of MILA (Cont..)

- Detector set in MILA is dynamic whereas detector set in Negative Selection Algorithm remains constant once it is generated in training phase.
  - The cloning, mutation and selection after detect phase in MILA is actually a process of on-line learning and optimization.
  - The process of cloning in MILA is a targeted (not blind) cloning. Only those detectors that are activated in recognition phase can be cloned.
  - This strategy ensures that both the speed and accuracy of detection becomes successively higher after each detecting.
Summary

- AIS emerged in 1990s as a new paradigm in AI, and has earned its position on the map of soft computing
- Being used in many applications – anomaly detection, pattern recognition, data mining, computer security, adaptive control, fault detection
- The long-term usefulness of AIS methods still depend on
  - Uniqueness
  - Effectiveness

We need unified AIS architecture and/or algorithm
For Information on Artificial Immune System Related Events and Bibliography

Visit the website
http://www.cs.memphis.edu/~dasgupta/AIS/

The University of Memphis