VARIABLE SHAPED DETECTOR: A NEGATIVE SELECTION ALGORITHM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ZAFER ATASER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

FEBRUARY 2013

Approval of the thesis:

VARIABLE SHAPED DETECTOR: A NEGATIVE SELECTION ALGORITHM

submitted by **ZAFER ATASER** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı
Head of Department, **Computer Engineering**

Prof. Dr. Ferda Nur Alpaslan
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Prof. Dr. Mehmet R. Tolun
Computer Engineering Department, TED University

Prof. Dr. Ferda Nur Alpaslan
Computer Engineering Department, METU

Prof. Dr. Müslim Bozyiğit
Computer Engineering Department, Maltepe University

Assoc. Prof. Dr. Tolga Can
Computer Engineering Department, METU

Assoc. Prof. Dr. Ahmet Çoşar
Computer Engineering Department, METU

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: ZAFER ATASER

Signature :

ABSTRACT

VARIABLE SHAPED DETECTOR: A NEGATIVE SELECTION ALGORITHM

Ataser, Zafer
Ph.D., Department of Computer Engineering
Supervisor : Prof. Dr. Ferda Nur Alpaslan

February 2013, 73 pages

Artificial Immune Systems (AIS) are class of computational intelligent methods developed based on the principles and processes of the biological immune system. AIS methods are categorized mainly into four types according to the inspired principles and processes of immune system. These categories are clonal selection, negative selection, immune network and danger theory. The approach of negative selection algorithm (NSA) is one of the major AIS models. NSA is a supervised learning algorithm based on the imitation of the T cells maturation process in thymus. In this imitation, detectors are used to mimic the cells, and the process of T cells maturation is simulated to generate detectors. Then, NSA classifies the specified data either as normal (self) data or as anomalous (non-self) data. In this classification task, NSA methods can make two kinds of classification errors: a self data is classified as anomalous, and a non-self data is classified as normal data.

In this thesis, a novel negative selection method, variable shaped detector (V-shaped detector), is proposed to increase the classification accuracy, or in other words decreasing classification errors. In *V-shaped detector*, new approaches are introduced to define self and represent detectors. *V-shaped detector* uses the combination of Local Outlier Factor (LOF) and kth nearest neighbor (k-NN) to determine a different radius for each self sample, thus it becomes possible to model the self space using self samples and their radii. Besides, the cubic b-spline is proposed to generate a variable shaped detector. In detector representation, the application of cubic spline is meaningful, when the edge points are used. Hence, Edge Detection (ED) algorithm is developed to find the edge points of the given self samples. *V-shaped detector* was tested using different data sets and compared with the well-known one-class classification method, SVM, and the similar popular negative selection method, NSA with variable-sized detector termed *V-detector*. The experiments show that the proposed method generates reasonable and comparable results.

Keywords: Negative selection, cubic spline, local outlier factor, detector coverage

ÖZ

DEĞİŞKEN ŞEKİLLİ DETEKTÖR: BİR NEGATİF SEÇME ALGORİTMASI

Ataser, Zafer
Doktora, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Ferda Nur Alpaslan

Şubat 2013 , 73 sayfa

Yapay Bağışıklık Sistemleri (YBS), biyolojik bağışıklık sisteminin ilkeleri ve süreçleri temel alınarak geliştirilmiş hesaplama akıllı yöntemler sınıfıdır. YBS yöntemleri, bağışıklık sisteminin ilham alınan ilke ve süreçlerine göre başlıca dört tür olarak kategorize edilir. Bu kategoriler klon seçme, negatif seçme, bağışıklık ağ ve tehlike teorisidir. Negatif seçme algoritması (NSA) yaklaşımı önemli YBS modellerden biridir. NSA, timusta T hücrelerinin olgunlaşma süreci taklidi üzerine kurulu bir denetimli öğrenme algoritmasıdır. Bu taklitte, hücreleri taklit etmek için detektörler kullanılır ve T-hücreleri olgunlaşma süreci detektörleri oluşturmak için simüle edilir. Sonra, NSA verilen veriyi ya normal (kendinden) ya da anormal (kendinden olmayan) olarak sınıflandırır. Bu sınıflandırma görevinde, NSA yöntemleri iki türlü sınıflandırma hatası yapabilir: normal veri anormal veya anormal veri normal olarak sınıflandırılır.

Bu tezde, sınıflandırma hatalarını azaltırken sınıflandırma doğruluğunu artırmak için değişken şekilli detektör (D-şekilli detektör) olarak adlandırılan yeni bir negatif seçme yöntemi önerilmiştir. D-şekilli detektör yönteminde normali belirleme ve detektör gösterimi için yeni yaklaşımlar tanıtıldı. D-şekilli detektör yöntemi, normal örneklerinin her biri için farklı bir yarıçap belirlemek amaçlı Yerel Aykırı Faktör ve k en yakın komşu yöntemlerini birlikte kullanmaktadır. Böylece, normal veriler ve yarıçapları kullanılarak normal uzayı modellenmek mümkün hale gelir. Ayrıca, değişken şekilli detektör oluşturmak için kübik spline yöntemi önerilmiştir. Detektör gösteriminde, kenar noktaları kullanıldığında kübik spline uygulama anlamlıdır. Böylece, verilen normal örneklerden kenar noktaları bulmak için kenar bulma algoritması geliştirilmiştir. D-şekilli detektör yöntemi test edilmiş ve iyi bilinen tek-sınıf sınıflandırma yöntemi (SVM) ve benzeri bir başka negatif seçme yöntemi karşılaştırılmıştır. Deneyler önerilen yöntemin makul ve karşılaştırılabilir sonuçlar ürettiğini göstermektedir.

Anahtar Kelimeler: Negatif seçme, kübik spline, yerel dış etmeni, detektör kapsama

To my son Salih

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my advisor Prof. Dr. Ferda Nur Alpaslan for her patience, tolerance and encouragement to complete my thesis. Besides my advisor, I would like to thank the rest of my thesis committee for their meaningful feedbacks and suggestions.

I am truly grateful to my friend Mustafa Yavaş for his motivation and support. He continually encourages me to complete my thesis study. Besides, I would definitely like to thank to my friend Seyit Sabri Seyhan for his support to pursue this degree. Also, thank to my friend Ali Çelikçekiç for his moral support.

Great thanks to my wife Şaziye Ataser, for her tolerance, when I need to study. She continually support and encourage me to finish my thesis study. I would like to thank my son to make me cheer when I was in pessimist mood. At last, I would like to express my gratitude to my mother, father and brothers for their full support through my education life.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xiii

LIST OF FIGURES xiv

LIST OF ALGORITHMS xvi

CHAPTERS

1 INTRODUCTION 1

2 RELATED WORK 5

  2.1 Major AIS Works 5

    2.1.1 Introduction to AIS 5

    2.1.2 Immune Network Theory 6

    2.1.3 Clonal Selection 7

    2.1.4 Danger Theory 8

    2.1.5 Applications 9

  2.2 Negative Selection Algorithm (NSA) 9

3 BACKGROUND 15

3.1 Negative Selection Algorithms 15

3.1.1 Self Definition 16

3.1.2 Detector Representation 17

3.2 One-Class Classification 20

3.3 *k* Nearest Neighbor 21

3.4 Local Outlier Factor 21

3.5 B-Spline 22

4 V-SHAPED DETECTOR: A NEGATIVE SELECTION ALGORITHM 27

4.1 Anomaly Detection 27

4.2 Self Space 28

4.3 Detector Representation 28

4.4 Edge Points Detection 31

4.5 V-Shaped Detector Algorithm 34

5 EXPERIMENTATION AND RESULTS 41

5.1 Comparison with SVM 46

5.2 Comparison with V-detector 49

5.3 Summary 56

6 CONCLUSION 57

REFERENCES 59

APPENDICES

A TEST METHODS AND PARAMETERS 65

A.1 Support Vector Machine (SVM) 65

A.2 Variable-Sized Detectors Negative Selection Algorithm (V-detector) 67

B TEST DATA SETS AND PREPROCESSING 69

B.1 1999 DARPA Intrusion Detection Evaluation Data Sets 69

B.2 Fisher's Iris Data Set 70

B.3 Biomedical Data Set 70

CURRICULUM VITAE 73

LIST OF TABLES

TABLES

Table 2.1 Application Areas of AIS [1]. 9

Table 5.1 Comparison between *V-shaped detector* and *oc*SVM using 1999 DARPA Intrusion Detection Evaluation data sets 47

Table 5.2 Comparison between *V-shaped detector* and *oc*SVM using Fisher's Iris data set . . . 48

Table 5.3 Comparison between *V-shaped detector* and *oc*SVM using Biomedical data set . . . 48

Table 5.4 Comparison between *V-shaped detector* and *V-detector* using 1999 DARPA Intrusion Detection Evaluation data sets 50

Table 5.5 Comparison between *V-shaped detector* and *V-detector* using Fisher's Iris data set . . 52

Table 5.6 Comparison between *V-shaped detector* and *V-detector* using biomedical data set . . 55

Table B.1 Second week attacks description. 70

LIST OF FIGURES

FIGURES

Figure 2.1 Generation of valid detector set [2]. 10

Figure 2.2 The lifecycle of a detector [3]. 11

Figure 3.1 General definitions for classification and NSA 16

Figure 3.2 (a) Small self radius. (b) Large self radius. 17

Figure 3.3 (a) Boundary-aware. (b) Variable self radius. 17

Figure 3.4 Examples of detectors' geometrical shapes. 18

Figure 3.5 The simplified detector generation process. 19

Figure 3.6 Illustration of one-class classification problem in two dimensional space. 20

Figure 3.7 Illustration of k-NN classification. 21

Figure 3.8 Illustration of LOF values of points. 23

Figure 3.9 First-degree spline function. 23

Figure 3.10 (a) Curve drawn by cubic spline with all points. (b) Curve drawn by cubic spline with edge points. 25

Figure 4.1 Self space and variable self radii for setos instances from Fisher's Iris data set. . . . 29

Figure 4.2 Variable shaped detectors. 30

Figure 4.3 Illustration of detectors that contain the self samples. 31

Figure 4.4 Input and edge points and their corresponding curves. 32

Figure 4.5 Traversing the neighbors of a point. 33

Figure 4.6 The result of Edge Detection algorithm. 35

Figure 4.7 General diagram of *V-Shaped detector* architecture. 36

Figure 5.1 The first and second week Monday data sets. 42

Figure 5.2 Reduced Fisher Iris Data. 42

Figure 5.3 Reduced Biomedical Data. 42

Figure 5.4 Relations between *MaxCosine*, average self radius, detection rate and false alarm rate. 44

Figure 5.5 ROC diagrams for Fisher's Iris data set. 45

Figure 5.6 ROC diagrams for 100%, 50% and 25% of training data sets of 1999 DARPA Intrusion Detection Evaluation data sets. 51

Figure 5.7 Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using 1999 DARPA Intrusion Detection Evaluation data sets. 53

Figure 5.8 ROC diagrams for 50% and 25% of training data sets of Fisher's Iris data set. 54

Figure 5.9 Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using Fisher's Iris data set. 54

Figure 5.10 Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using biomedical data set. 55

Figure 5.11 ROC diagrams for V-detector and V-shaped detector using biomedical data. 55

Figure B.1 Block diagram of 1999 DARPA test bed [4]. 69

Figure B.2 Output of tcpstat for the first week's outside tcpdump. 71

Figure B.3 Reduced Fisher Iris Data 71

Figure B.4 Reduced Biomedical Data 72

LIST OF ALGORITHMS

ALGORITHMS

Algorithm 1 Pseudocode of the Negative Selection Process to generate detectors [5] 11

Algorithm 2 Local Outlier Factor (LOF) algorithm 22

Algorithm 3 Edge points detection algorithm for two-dimensional input 34

Algorithm 4 Negative Selection with B-Spline (V-shaped detector) 36

Algorithm 5 Monte Carlo algorithm to calculate the estimated self volume 37

CHAPTER 1

INTRODUCTION

Artificial Immune Systems (AIS) are a category of computational intelligent models inspired by the principles and processes of the biological immune systems. Various models of AIS were developed using different biological immune principles and processes. These AIS models are categorized mainly into four types: immune network model, clonal selection, negative selection and danger theory [6].

The important issue in AIS is the correct discrimination between self and non-self. To achieve this discrimination, negative selection method was designed by modeling the generation of T-cells in thymus. In the biological process, T-cells are generated in thymus and each one is checked whether it is bounded self-cells or not. If it is bounded, then it is eliminated. This process is repeated until the T cell is maturated, and then the maturated T cells are released to blood.

In negative selection method, units called detectors are used instead of T-cells to discriminate normal (self)[1] data or not (non-self)[2] data [2]. The main concern of a collection of detectors, in other words detector set, is the maximization of the non-self space. Detection rate directly depends on the proportion of the non-self space covered by the detector set.

In negative selection algorithm (NSA), there are two main phases, detector generation and detection. In the generation step, candidate detectors are randomly generated and then eliminated in case of matching self samples. The candidates that do not match self samples constitute the detector set. In the detection step, the incoming instance is checked whether it matches any detector or not. If it matches, then it is a non-self or anomaly.

NSA classifies an input data either as normal (self) data or anomalous (non-self) data in the detection part. Hence, it can be easily said that NSA is proper for one-class classification problems such as fault detection, and anomaly detection [7]. Although, there are many fields that use NSA, the primary application of NSA has been anomaly detection. The purpose of anomaly detection is also the discrimination of normal and anomalous state of a system.

Positive and negative terms should be described before mentioning the classification accuracy of NSA. A positive term states the detection of anomalous (non-self) data while a negative term defines the detection of normal (self) data. The positive and negative terms are divided into two groups; a true shows correct detection and a false shows incorrect detection.

In detection process of NSA, the algorithm can make two types of errors. A false positive defines the detection of self-data as anomalous, and a false negative defines the detection of non-self data as

---

[1] Normal and self are used interchangeably in this thesis.

[2] Anomalous and non-self are used interchangeably.

normal data. Detection accuracy directly depends on these two types of errors. Therefore minimizing these errors increases detection correctness.

To minimize the false positive and the false negative errors, the self region determination and the non-self space coverage by detectors are critical issues. Self region is defined by self samples and their radius. Mostly, self samples' radius is selected as constant and it is given as the parameter. On the other hand, some NSAs find different radius for each self sample using different methods.

The detector representation and matching rule affects the non-self space coverage by detectors. Most of the NSAs use binary encoding representation and r-contiguous bits as matching rule; two strings match if they have r contiguous bits in common [3]. Instead of using binary encoding, some of the negative selection variants use real-valued representation to characterize the self/non-self space and generate a set of detectors that try to cover the (non-self) complementary subspace [8].

Real valued negative selection algorithms use a specific geometrical shape - rectangle, circle and so on - or a mixture of geometrical shapes as a detector. Negative selection generates many detectors to cover non-self space. Due to the specific shapes of detectors, these algorithms have a limitation to cover non-self space.

This study is motivated from the study of intrusion detection that uses NSA, and is developed to enhance NSA's classification accuracy. Although, the performance of existent NSA is reasonable, it is possible to enhance NSA classification. Hence, this paper proposes a new real valued negative selection algorithm in order to maximize classification accuracy while minimizing classification errors. To that end, it is focused on two issues, self space determination and non-self coverage. In this new NSA, local outlier factor (LOF) and kth nearest neighbor (k-NN) are used to determine self boundary. Furthermore, generated detectors' shapes are generally circles. However, some detectors' shapes can be changed when they touch the self-data. To make variable shaped detectors, cubic b-spline interpolation method is used. The cubic b-spline finds function of the given self-data which is covered by the detectors. In this way, these functions determine the boundaries of detectors.

The main contribution of this study was implemented with new negative selection algorithm, called *V-shaped detector*. There are several important features of *V-shaped detector* that distinguish it from others and handle some issues in negative selection algorithms.

- *V-shaped detector* adds some properties to detectors to obtain maximum non-self coverage. Detectors are randomly generated as variable sized circles and the proposed algorithm allows detectors to contain limited number of self samples. Shapes of detectors that contain self samples are changed based on the covered self samples using cubic spline. Thus, the algorithm uses variable sized and variable shaped detectors. This is the first research to use variable shaped detectors to cover non-self space. This property gives opportunity to minimize holes between self space and covered non-self space.

- Self space determination is the other core issue in negative selection algorithms, because it directly affects accuracy of detector coverage. Self adaptability is proposed to determine more accurate self space. In order to obtain self adaptability, local and global features are combined. The local feature is extracted using Local Outlier Factor, a density based outlier method. The global feature is extracted using the kth nearest neighbor method. *V-shaped detector* does not need prior knowledge about input data with the use of Local Outlier Factor and kth nearest neighbor. This is the first research to use the combination of Local Outlier Factor and kth nearest neighbor for self adaptability.

- *V-shaped detector* algorithm was needed a method to find edge points of self samples, so it can apply cubic spline to form variable shaped detectors. Hence, edge detection algorithm was developed as a component of *V-shaped detector*. This new algorithm can be used in other disciplines, i.e. image processing, to extract a shape of input.

- Negative selection algorithms generally use iterative or evolutionary process to generate detectors, but *V-shaped detector* uses modified Monte Carlo Integration method, a type of estimation method, to generate detectors. In this modified Monte Carlo Integration method, randomly generated detectors are not discarded, if they do not satisfy the detector rules. Instead of detectors discard, their attributes, against the rules, are changed to obey the detector rules. This method speeds up detector generation process in *V-shaped detector* algorithm.

The proposed NSA was evaluated using two important data sets, 1999 DARPA Intrusion Detection Evaluations data sets and Fisher Iris Data. The proposed NSA was developed as a new approach for one-class classification, meaning a two-class classification that can be trained with samples from one of the classes. Hence, each data set was divided into training and test data. The proposed NSA and the known NSA, *V-detector*, were evaluated based on detection rate, false alarm rate, and the number of detectors.

A part of these studies has been published in [9] and other part are ready to be published. Therefore, the significant parts of the explanations in this and the following sections are taken from these publications.

The next chapters of this thesis are organized as follows. First, related work is introduced in Chapter 2. Background information about NSAs is presented in Chapter 3, and the proposed new NSA is explained in Chapter 4. The experiments and results that are generated by the proposed NSA and *V-detector* are evaluated in Chapter 5. The conclusion remarks are given in Chapter 6.

CHAPTER 2

RELATED WORK

This chapter reviews the studies that are related to the background of the proposed method. These studies are explained in the following two sections: Section 2.1 presents some major works about all fields of AIS and Section 2.2 reviews negative selection algorithms in detail.

2.1 Major AIS Works

Artificial Immune System (AIS) attracted the attention of researchers after the first studies, and many researches have been done on it. AIS emerged as a new branch of Artificial Intelligence (AI) as other disiplines inspired from biological mechanisms. There are many studies on AIS, and some of them focused on the categorization of the proposed AIS methods based on their porperties [6],[7]. This section reviews the studies on existing models.

2.1.1 Introduction to AIS

The biological mechanisms have always been the main source for the development of new computational models to solve problems. The biological immune system is one of them, and researches inspired by the biological immunity caused the emergence of a new area, Artificial Immune System (AIS), for computational intelligence.

The biological immune system is still an active research area for the biology discipline, and as a result of the researches, many theories have been proposed about the mechanism of the biological immune system. Hence, various models of AIS were developed inspired by the different biological immune theories. Garret [6] present these models as immune network, clonal selection, negative selection and danger theory.

The immune system constitutes of two main mechanisms, innate immunity and adaptive (acquired) immunity. The innate immunity is the first layer defense mechanism, and the second one is the adaptive immunity. The innate immunity consists of basic elements the organism is born with such as skin and physical barriers. The adaptive immunity provides the ability to adapt over time to recognize specific pathogens, and it creates immunological memory after an initial response to a specific pathogen. The clonal selection theory was developed to describe the principles of adaptive immunity. The main features of the clonal selection theory are [10]:

- Clone activated mature cells and generates random changes on clone cells with high rates (somatic mutation);

- Elimination of newly differentiated cells which match self cells;
- proliferation and differentiation on activation of cells by antigens.

The immune network theory was developed to explain the adaptive immune system mechanism, and it had been introduced by Jerne [11]. The theory states that the immune system maintains an idiotypic network of interconnected B cells for antigen recognition. These cells interact with each other, and they interconnect with each other in definite rules to stabilize the network. Two interacting cells are connected if their affinities exceed a certain threshold. The strength of the connection is directly proportional to their affinity [7].

The biological negative selection describes T cells maturation process in thymus called T cell tolerance. T cell's gene segments are randomly rearranged together by somatic gene rearrangement, and bases are inserted to create T cell against antigens. The generated cells are eliminated when they recognize self cells as antigens. In the end of this elimination process, the remaining cells, mature cells, are released from the thymus. In this manner, these mature cells increase the ability of the immune system to detect unknown antigens. Inspired by the biological negative selection, the process of negative selection generates a set of T-cell detectors that can detect any form of non-self in AIS [6].

The main idea of danger theory states that the immune system responds to danger instead of non-self [12]. Danger theory fundamentally accepts the need for discrimination as other theories. On the other, the difference between danger theory and others is the answer to what should be responded to. Danger theory responds danger instead of foreignness, and danger is evaluated by damage to cells specified by distress signals. These signals are sent out in case of an unnatural death of cells.

2.1.2 Immune Network Theory

Timmis et al. [13], [14] introduced Artificial Immune Network (AINE) method which uses artificial recognition ball (ARB) to represent a number of identical B cells described in immune network. The stimulated B cells are subjected to clone and somatic hypermutation ensures that these clones are differentiated a relatively large proportion of the parent cell. The cells stimulated by particular antigen are kept in the immunological memory for that antigen. Two B cells are connected based on the affinity between them. The affinity is measured using the Euclidean distance between the two B cells. The two B cells are connected when the affinity between them exceeds the network affinity threshold. The connected B cells are called as ARB. Timmis et al. [14] were tested AINE using Fisher Iris dataset, and AINE generates the disconnected clusters which provide the variety which allows the immune system to generalize variations in the data encountered.

Castro and Zuben [15] proposed "Artificial Immune Network Model for Data Analysis" (aiNet) learning algorithm. This algorithm uses nodes simulating antibodies, while AINE uses nodes inspired by B cells. Antibodies are receptor molecules that are secreted B cells with the primary role recognizing and binding with an antigen. One of the important aims of this algorithm is to increase the generalization of antibodies and the network. In order to do that, aiNet suppresses antibodies with low antigenic and high affinities according to the suppression threshold. There are two suppressive steps in this algorithm, clonal suppression and network suppression. Hence, the suppression threshold controls the specificity level of the antibodies, the clustering accuracy and network plasticity. aiNet provides the reconstruction of the metric and topological relationships. Reproducing the topological relationships causes that similar information are mapped onto closer antibodies, eventually the same one and clustering of the input space.

Liu and Xu [16] introduced a cooperative artificial immune network called CoAIN to improve search ability and search speed. The CoAIN uses cooperative strategy inspired by particle swarm behavior. This means that each network cell has the ability to cooperate with other individuals, and this cooperation adjusts position according to its own experience and the experience of the best cell. In this way, this cooperation ability finds the best position encountered by itself and its neighbor. In the other feature of CoAIN, antibodies with fitness dominate clonal selection, and smaller step size of mutation is used in clonal selection to find global optimization. Step size of mutation is decreased smoothly with the increase of generation to fit for finer search. This paper shows that some basic immune principles together with simple cooperation behavior makes possible to solve complex optimization tasks.

Coelho and Zuben [17] proposed Concentration-based Artificial Immune Network (cob-aiNet) to solve single optimization problems, and they [18] introduce the extension of cob-aiNet to solve multi-objective optimization problems. The cob-aiNet exploits the features of a concentration-based immune model. Thus, it controls the dynamics of the population, and uses new mechanisms to stimulate and maintain the diversity of the individuals in the population.

Zhong and Zhang [19] present a novel supervised algorithm based on the immune network theory, the artificial antibody network (ABNet). In this method, every antibody consists of two important attributes, its center vector and recognizing radius. The antibody can recognize all antigens within the range of its recognizing radius. ABNet was designed for classifications of multi-/hyperspectral remote sensing images.

2.1.3 Clonal Selection

Castro and Zuben [20], [21] popularized the artificial form of clonal selection developing an algorithm called clonal selection algorithm (CSA). They applied CSA to different problems and compared it with the standard genetic algorithm (GA) [21]. Then, they [10] modified the algorithm and renamed it to CLONALG, which is the most known clonal selection algorithm. Two forms of CLONALG were introduced, one for optimization tasks and one for pattern matching. CLONALG takes into account the following main immune features;

- maintenance of a specific memory set;
- selection and cloning of the most stimulated antibodies;
- death of nonstimulated antibodies;
- affinity maturation
- reselection of the clones according to their antigenic affinity, generation, and maintenance of diversity.

Brownlee [22] introduced Clonal Selection Classification Algorithm (CSCA) which is mainly based on CLONALG. Concern of CSCA increases classification accuracy. CSCA is considered as a function optimization procedure that maximizes the number of correctly classified patterns and minimizes the number of misclassified patterns. CSCA is considered as a function optimization procedure that maximizes the number of correctly classified patterns and minimizes the number of misclassified patterns. The algorithm is constituted of four main steps:

1. Initialization - Initialize the antibody population.
2. Training looping - Involve selection and pruning step, cloning and mutation step and the insertion the generated clones.
3. Final Pruning - Prepare fitness scores and perform pruning
4. Classification - Classify using the antibody population.

Oliveira et al. [23] proposed Clonal Selection Classifier with Data Reduction (CSCDR) which is mainly based on the CSCA. CSCA is modified to increase the performance and to decrease the number of memory cells. The mutation process is changed to get better results in search space process, and a control parameter is inserted to decrease the number of memory cells produced.

Li et al. [24] introduced Reconfigurable Space Clone Selection Algorithm (RSCSA), which is focuses on the antibody population size and antibody search space. RSCSA reduces the search space and antibody population size. Due to this reduction, the algorithm has strong robustness and fast convergent speed.

2.1.4 Danger Theory

Aickelin and Cayzer [32] presented the one of the first major studies about the danger theory from AIS perspective. This study explains Matzinger's Danger Theory in the first part. Then, the danger theory is evaluated from the perspective of AIS practitioners. In this evaluation, the danger concept is discussed, and the danger theory is compared with the other AIS models, i.e. negative selection. Beside this, they debate about how to implement the danger theory based on the AIS perspective. In the last section, the AIS applications are evaluated based on the danger theory.

Greensmith et al. [33] described the danger theory and AIS applications. The current state of intrusion detection systems (IDS) was presented. They discussed the application of the danger theory on IDS, and claimed that significant improvements will be provided. Kim et al.[34] and Roper [35] also proposed that the danger theory is more suitable than other AIS models to apply on IDS.

Aickelin and Greensmith [36] introduced two algorithms, the Dendritic Cell Algorithm (DCA) and the Toll-like Receptor algorithm (TLR), developed based on the danger theory. These algorithms were developed inspired by different aspects of the danger theory. DCA and TLR proved that it is possible to build feasible AIS algorithms based on the principles of the danger theory.

Zhu and Tan [37] proposed a danger theory based learning (DTL) model, which mimic the mechanism of the danger theory. The algorithm was tested with spam filtering problem and compared with classical machine learning approaches, Support Vector Machine (SVM) and Naive Bayes (NB). In experiments, the DTL model outperformed SVM, NB.

Table2.1: Application Areas of AIS [1].

Major	Minor
Clustering/Classification	Bio-informatics
Anomaly Detection	Image Processing
Computer Security	Control
Numeric Function Optimisation	Robotics
Combinatoric Optimisation	Virus Detection
Learning	Web Mining

2.1.5 Applications

AIS have many application areas today after first proposal. Hart and Timmis [1] surveyed AIS studies and classified application areas of AIS into 12 headings. These categories are presented in table, and according to the number of researches on these categories, categories were divided into major and minor. Based on these categorizations of application areas, Hart and Timmis summarized application areas of AIS as (1) Learning (2) Anomaly Detection and (3) Optimisation. Application areas were mapped to these groups: Learning contains clustering, classification and pattern recognition, robotic and control applications; Anomaly Detection includes fault detection and computer and network security applications; Optimisation consists of real-world problems which essentially include combinatoric and also numeric function optimisation.

Garret [6] surveyed AIS models and gives the application areas for each of them. NSA application areas are change detection, fault detection and diagnosis, network intrusion detection; Clonal selection application areas are pattern recognition, automated scheduling, document classification, uni-modal, combinatorial and multi-modal optimization; Immune network application areas are detecting gene promoter sequences, diagnosis data mining and cluster analysis; Danger theory application area is intrusion detection.

Freitas and Timmis [5] discussed the application of AIS for data mining. They evaluated all AIS models and found limitations in existing AIS for data mining. Limitations they discovered and suggestions for future researches were mentioned in order to mitigate corresponding limitation.

2.2 Negative Selection Algorithm (NSA)

Artificial Immune System (AIS) covers many models inspired by the biological immune system. The first model, negative selection algorithm (NSA), among AIS models was introduced by Forrest et al. [2]. Many researches have been performed after the introduction of NSA. These researches proposed various NSA, and they are differentiated in data representation, detector representation, self definition and matching rule.

Forrest et al. [2] proposed NSA inspired by discrimination between self and non-self in immune system. Therefore, NSA imitates the T cell maturation process, which gives the ability to T cells to make discrimination between self and non-self. To implement NSA, there are some critical development considerations; data representation, self definition, detector coverage, matching rule. This study converts the given data to binary representation, so detectors are also represented in binary form. Self was defined as the string to be protected, and other (non-self) to be any other string. The self string

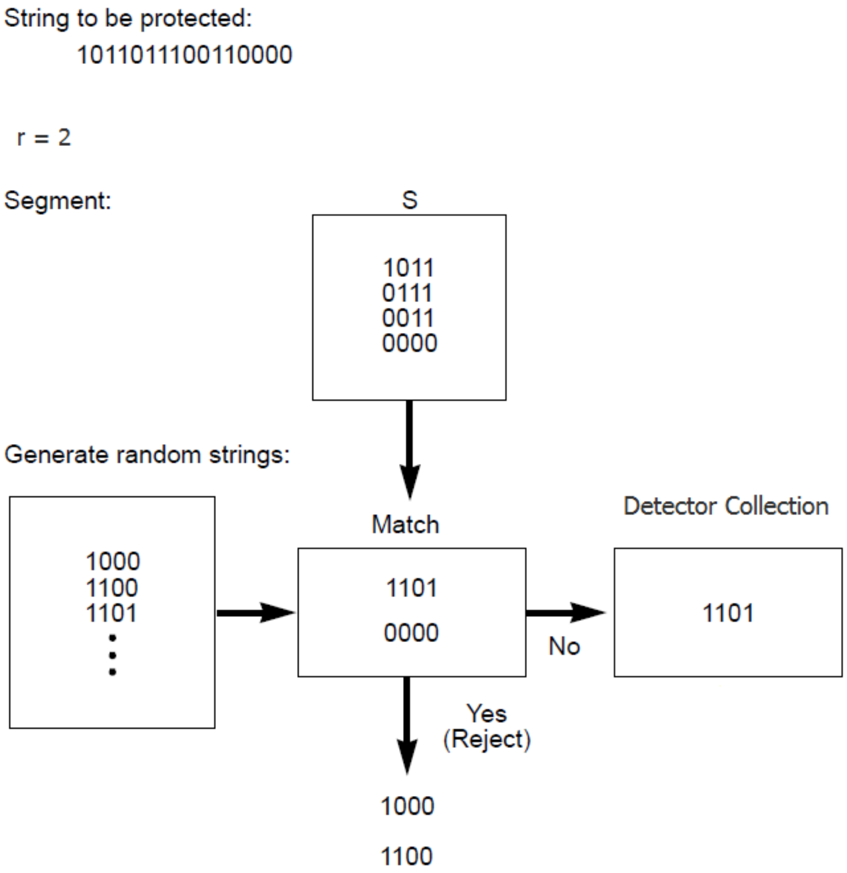


Figure 2.1: Generation of valid detector set [2].

is logically split into equal-size segments to generate valid detectors. This produces the collection *S* of self substrings. In the second step, detectors are randomly generated, and generated strings match strings in *S* are eliminated. Strings that do not match any string in *S* are added to the detector set. Two strings match, if they match at least *r* contiguous locations. Figure 2.1 shows all these the detector generation phase.

Freitas and Timmis [5] discussed the application of NSA for data mining, so features of NSA were explored. The basic framework of the negative selection process to generate detectors were given in

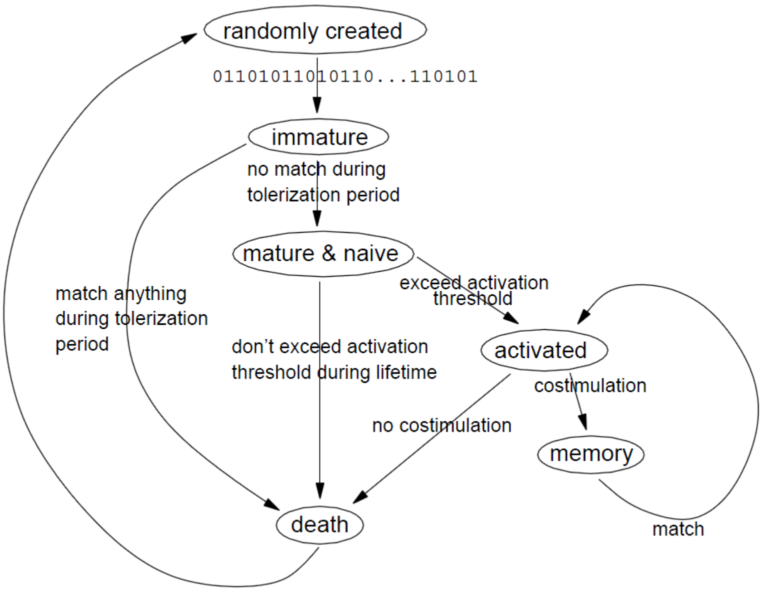


Figure 2.2: The lifecycle of a detector [3].

Algorithm 1.

Algorithm 1: Pseudocode of the Negative Selection Process to generate detectors [5]

```
Data: a set of normal (self) data instances ($S$)
Result: a set of mature detectors that do not match any instance in $S$
repeat
    Randomly generate an immature detector
    Measure the affinity (similarity) between this detector and each instance in $S$)
    if the affinity between the detector and at least one instance in $S$ is greater than a
    user-defined threshold then
        discard this detector
    else
        output this detector as a mature immune detector
until stopping criterion;
```

Hofmeyr and Forrest [30],[3] proposed the artificial immune system (ARTIS) method, and they applied it to intrusion detection. This method represents detectors as bit strings, and uses the r contiguous matching rule. Beside these, the study defines the lifecycle of a detector, so it provides dynamic detector populations and adaptation ability in a continuously changing environment. In this lifecycle, a detector can be in one of the five states: immature, mature, activated, memory or death. Figure 2.1 presents the lifecycle of a detector. Randomly generated detector is considered as immature detectors, and if it does not match self data during the tolerization period, it becomes a mature detector. A mature detector becomes an activated, when it exceeds the activation threshold (match threshold). After that, a human security officer's confirmation (costimulation) is needed for an activated detector to make it a memory detector. Immature, mature and activated detectors can die, but memory detectors go to activated state, when they match non-self. An immature detector dies, if it matches self. A mature detector death is occurred, when it does not exceed activation threshold during lifetime (life expectancy). An activated detector dies, if it does not receive confirmation in a time period (costimulation delay).

Gonzalez et al. [25] presented the effects of the low-level representation and its matching rules on the performance of NSA in covering the non-self space. They explored and compared the different binary matching rules: r-contiguous matching, r-chunk matching, Hamming distance matching, and Rogers and Tanimoto matching. This study indicates that the matching rule for NSA needs to be chosen when it represents data accurately in problem space.

Gonzalez et al. [26] proposed a Real-Valued Negative Selection (RNS) algorithm. RNS algorithm uses real numbers to represent self/non-self space RNS algorithm and binary NSA were compared for anomaly detection problem. Then, advantages and disadvantages of the real-valued representation were presented based on the binary representation. Real-valued representation advantages are: closer to original problem space, allowing the use of methods from computational geometry to speed-up the algorithms, facilitating the use of other machine learning methods to find useful high level knowledge i.e. [38]. Disadvantages of real-valued representation are: making analysis of the problem space harder, not suitable for the representation of categorical attributes.

Dasgupta and Gonzalez [8] explored positive selection and negative selection, and they were compared using real-valued representation. Detectors are represented as rectangle with real numbers. Based on this comparison, advantages and disadvantages of these approaches were described. This comparison showed that positive selection is more precise, but it needs more time and space resources. The negative selection is less precise, but it needs fewer time and space resources.

Real-valued representation is used in many applications due to the nature of applications' domains, i.e. intrusion detection from network traffic. The non-self coverage gets difficult for the problems with natural real-valued representation. This is because, the real-valued space is continuous and the boundary of self and non-self is ambiguous in this space. Therefore, the non-self coverage is a major issue for real-valued NSA (RNSA) [26], [27], [39], [40], [41], [42]. Detector representation and self definition are the determinant for the non-self coverage. A part of research have been focused on the detector representation and distribution in the non-self space in order to maximize the coverage[27], [41],[39]. On the other hand, the recent research is focused on adaptive-self that implicates the variable self radius [43], [40], [42] . The self radius is an important value to control the detection rate and false alarm rate.

In real-valued NSAs, the detectors are usually represented as circles or rectangles for two dimensional problems. Nevertheless, some NSAs use mixture of specific geometrical shapes to represent the detectors. However, some of NSAs generate the detectors with different sizes. Based on the data and detector representation, the matching rule is changed, and Euclidean distance matching is usually used in real-valued representation.

Dasgupta and Gonzalez [8], [44] represent the detectors generated by genetic algorithm as rules. They present the general form of detector rules (detectors) as follows:

$$R_j\text{: If } Cond_i \text{ then nonself, j} = 1,...,x$$

$$Cond_i = x_1 \in [low^i_1, high^i_1] \text{ and ... and } x_n \in [low^i_n, high^i_n]$$

where $(x_1,..., x_n)$ is a feature vector, and $[low^i_1,high^i_1\ ]$ specifies the lower and upper values in the condition part. In this definition, m is the number of detector rules, and n is the number of feature dimensions. The detectors correspond to hyper-rectangles in a multidimensional space. In this study, self region is determined by the level of variability (v) parameter, which is interpreted as the radius of self samples.

Gonzalez et al. proposed [27] a Randomized Real-Valued Negative Selection Algorithm (RRNS). This algorithm takes the detector radius and the self variability threshold (self sample radius) as parameters, so each self sample and detector is represented as circles in two-dimensional problem space. These circles have a fixed size specified by the relevant parameter. Based on the self radius parameter, the algorithm uses Monte Carlo method to estimate the volume of self region.

Balachandran et al [41] present a work focused on developing a framework for generating multi-shaped detectors in real-valued NSA. This new extended real-valued NSA uses multiple shape (sphere, rectangle or ellipse) detectors for covering two dimensional non-self space. In this NSA, self space is also specified by the constant self radius parameter.

Ji and Dasgupta [39],[45],[46],[47] proposed a new real-valued NSA, which generates variable size detectors. In this NSA, the detectors are represented as circles in two dimensional space and the radii of these circles are variable. On the other hand, the radius for all self samples is taken as the constant parameter and used to check whether a new generated detector is in any self circle or not. If it is, then discarded, otherwise the distance between the center of detector and the nearest self sample is assigned to this detector radius. This is called boundary-aware method [46].

In the work by Bezerra et al [43], an adaptive radius immune algorithm (ARIA) was developed. This is one of the first researches on variable self radius for each self sample. Although, ARIA is closer to clonal selection algorithm, this adjusted self radius has crucial effect in AIS. ARIA considers the density information to form its representation. ARIA takes an initial value of self radius and based on the local density of samples, this initial value is adjusted for each sample.

Zeng et al [40] introduce a self-adaptive negative selection algorithm (ANSA). ANSA can adapt the varieties of self/nonself space by adjusting self radii and detectors' radii. Yuel et al [42] worked on optimization of self set for real-valued NSA. In order to do that, self samples are processed in three steps. In the first step, wrong samples are discarded according to "3 σ" criterion. In the next step, the self radius is adjusted by the self's probability density. In the last step, unnecessary self samples, whose covered region is already overlapped by others, are discarded.

The major characteristics of a negative selection algorithm can be identified as follows:

1. Negative representation [48]: NSA identifies and represents the complementary space of the given samples in training phase. Negative representation and positive representation algorithms have been compared in many researches to extract the strength and applicability of negative representation [49], [8], [50], [51].

2. Usage of detector set as the classification mechanism [48]: Detector set usage provides the opportunity to NSA to distribute its processes, i.e. detectors generation.

3. One-class classification [48][5]: NSA was developed inspired by the self/non-self discrimination mechanism of the biological mechanism. Therefore, NSA is trained with samples from the one class (self) and then classifies the given instance into one of two classes (self/non-self). Although some researches tried to extend NSA for multiclass classification problems[8, 44], large majority of the researches have been applied to one-class classification problems.

4. Adaptation capability[30], [3] [52], [53],[54]: There are many researches to develop adaptive NSAs inspired by the adaptation ability of biological immune system. These adaptive NSAs use some mechanisms, i.e. memory, and processes, to obtain dynamic change of the detectors population.

Particularly, NSA was developed for intrusion detection research [2]. Negative selection algorithm (NSA) can be used in many domains today, but the most natural application domain of NSA is intrusion detection [8], [28], [29], [30], [31].

CHAPTER 3

BACKGROUND

Negative selection algorithm (NSA), especially real-valued NSA, is the one of the most popular AIS models interested by the researchers. There are a great many studies to increase NSA classification accuracy, while some of other studies focus on the comparison NSA with other related machine learning methods such as one-class classification methods to present the advantages of NSA. In this study, a new real-valued NSA is introduced, so the studies about real-valued NSA are in the scope of next subsections. The studies about the increasing classification accuracy mainly focus on two concepts of NSA, self definition and detector representation. Following subsection presents detailed information about NSA according to self definition and detector representation aspects. The later subsections give information about one-class classification, *k* nearest neighbor, Local Outlier Factor and cubic spline. These concepts and methods are related with this study, so information about them is given in next subsections.

3.1 Negative Selection Algorithms

Negative selection method models non-self space using self samples; whereas positive selection method models self space using self samples. Both methods classify a given data either as normal data or anomalous data and raise an alarm for an anomalous case.

In Dasgupta and Gonzalez [8], negative selection and positive selection were compared. This comparison showed that positive selection is more precise, but it needs more time and space resources. The negative selection is less precise, but it needs fewer time and space resources [8].

In order to mention the classification accuracy of NSA, positive and negative terms should be defined. A positive term means the detection of anomalous (non-self) data while a negative term means the detection of normal (self) data. Based on these definitions, classification results are categorized into four groups:

- A true positive (TP) occurs when non-self data is detected as anomalous data.
- A false positive (FP) occurs when self data is detected as anomalous data.
- A true negative (TN) occurs when self data is detected as normal data.
- A false negative (FN) occurs when non-self data is detected as normal data.

The false positive and false negative represent the types of classification errors. NSA should minimize them to increase the classification accuracy. Figure 3.1 represents the general definitions for

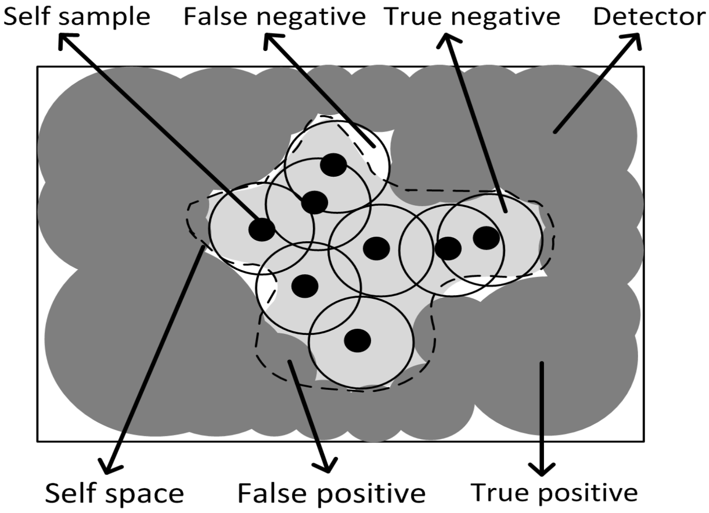


Figure 3.1: General definitions for classification and NSA

classification and NSA.

NSA is suitable for one class classification problems which means that each given data point belongs to one of two classes. Support Vector Machine (SVM) is the well-known method for one class classification problems. In Ji [55], SVM and *V-detector* were compared. SVM needs extra knowledge to choose the proper kernel function used to classify data. Besides this, it has some limitations for very large training data and discrete data. On the other hand, NSA does not need prior knowledge about the data and does not have any limitations for very large training dataset and discrete data [55].

3.1.1 Self Definition

The main factors that affect the classification accuracy of NSA are the self space determination and the non-self space coverage. The self radius has crucial effect on the self space determination and the self radius is given as a parameter almost in all NSA, i.e. [8], [45], [41]. Figure 3.2 illustrates the self radius selection effect on the self space determination. Figure 3.2(a) shows the self samples with small radius and Figure 3.2(b) shows the self samples with large radius.

In Figure 3.2(a), the self radius is too small and self samples cannot cover sufficient self space. In this case, high false positive rate occurs. However, selecting it too large can also be problematic. Figure 3.2(b) illustrates that the self radius is too large and part of the self samples covers the non self space. This causes high false negative errors.

V-detector, a well-known NSA develops a boundary-aware method to specify the self space using self samples [46]. This method takes self radius (r_s) as a parameter and uses it in detector generation phase. A generated detector is checked whether the distance between its center and a nearest self sample is greater than r_s. If not, then the generated detector is discarded. Otherwise, the distance is assigned to the radius of the generated detector. This means that if a detector center is outside of the self circle

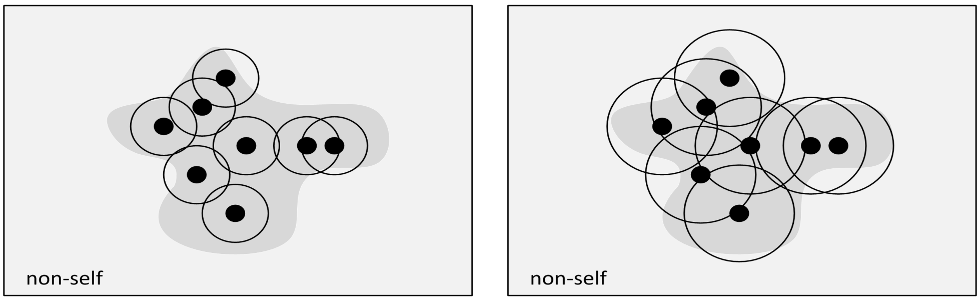


Figure 3.2: (a) Small self radius. (b) Large self radius.

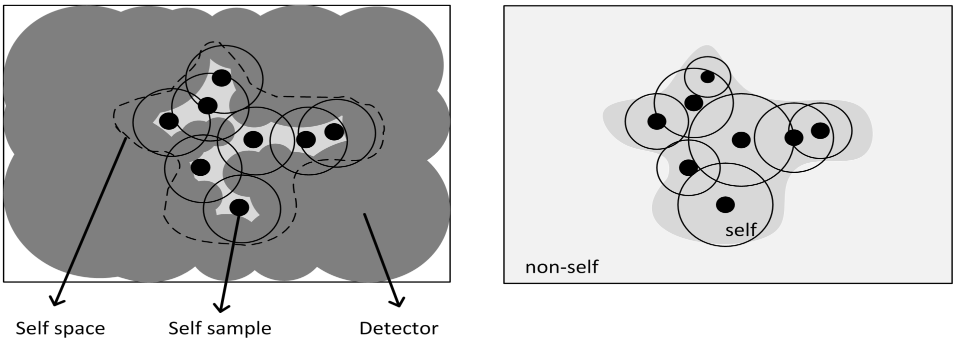


Figure 3.3: (a) Boundary-aware. (b) Variable self radius.

space, then it is valid detector and can cover a part of self circle space except self circle center namely self sample. Figure 3.3(a) illustrates the detected self space and generated detectors by *V-detector* with boundary-aware method. Nevertheless, this method does not take into account self features in the self space determination.

The recently developed NSAs calculate each self sample's radius using probabilistic techniques to find more accurate self space for the given self samples [40],[42]. Thus, a radius of each self sample is specified by the given samples' features and this provides self adaptability to NSA. Figure 3.3(b) presents the variable self radii which give opportunity to optimize the maximization of self space coverage and the minimization of non-self space coverage.

3.1.2 Detector Representation

The detector representation affects the non-self space coverage. Many NSAs represent detector using real-values [8], [27], [39], while others use binary encoding to represent detectors [2],[3]. Obviously there are two reasons to use binary representation. Binary representation constitutes a finite space that makes the analysis of the problem space easier. Besides this, binary representation is proper to represent categorical attributes [47]. The original problem space is usually continuous and real valued representation is more proper for these problems, i.e. intrusion detection [28]. Real valued

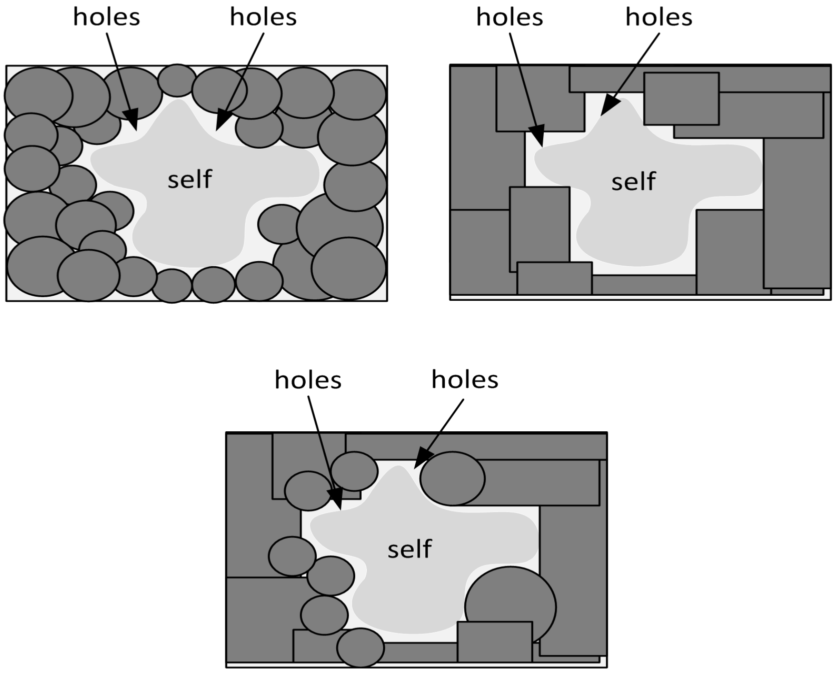


Figure 3.4: Examples of detectors' geometrical shapes.

representation provides to use computational geometry to speed-up the algorithms. Beside this, it also gives opportunity to apply other machine learning techniques to find useful high level knowledge.

Some real-valued negative selection algorithms represent detectors as a specific geometrical shape, i.e. circle [39], rectangle [8]for two dimensional problem space. On the other hand, some of them use a mixture of geometrical shapes to represent detectors [41]. Figure 3.4 shows circle, rectangle and a mixture of circle and rectangle detectors respectively. Due to specific shapes of detectors, these algorithms have limitation to cover non-self space. Therefore, there are holes, uncovered non-self space, between detectors and self space. These holes indicate classification errors.

In simplified detector generation process, NSA randomly generates a detector and checks whether it matches self or not. If it does not match self, it is accepted as valid detector otherwise it is regenerated and checked again. Figure 3.5 shows the simplified detector generation process. In some researches, a detector which matches self is modified according to matching self instead of the detector regeneration [46].

Matching rule is used in detector generation and detection processes. In detector generation, candidate detector is checked to see whether it matches with any self-data using matching rule. If it matches, then it is regenerated; otherwise, it is added to detector set. On the other hand, in detection processes, test data is classified according to the matching rule as a self data or a non-self, anomalous data.

Matching rule directly depends on the detector representation. Although there are many matching rule in real-valued representation, Euclidean distance based matching rule is usually used especially

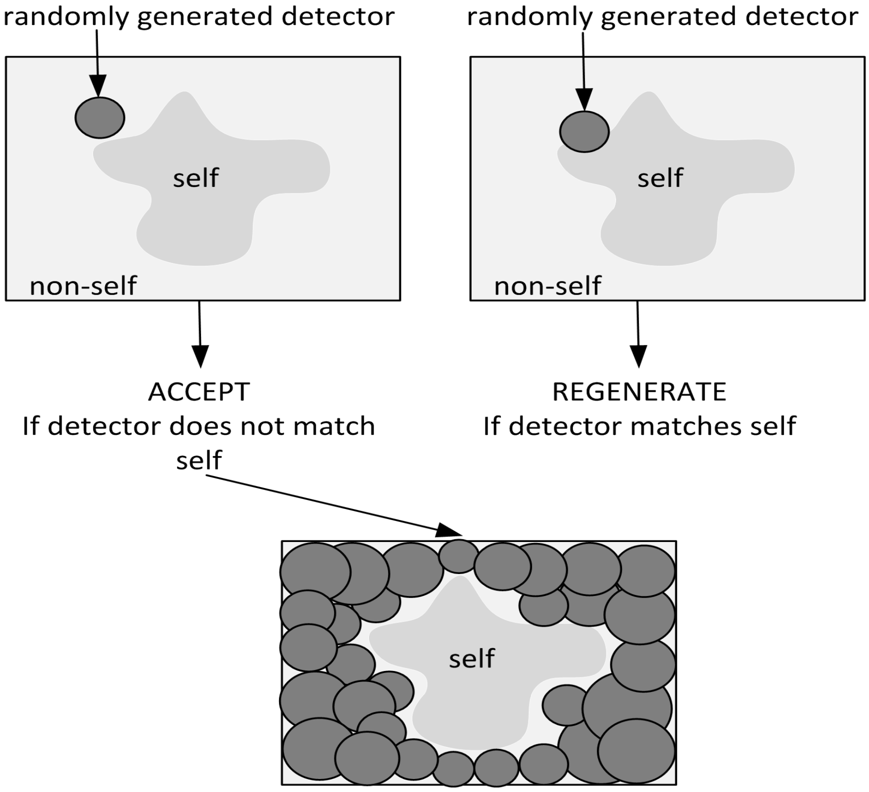

randomly generated detector
randomly generated detector
self
self
non-self
non-self
ACCEPT
If detector does not match self
REGENERATE
If detector matches self
self

Figure 3.5: The simplified detector generation process.

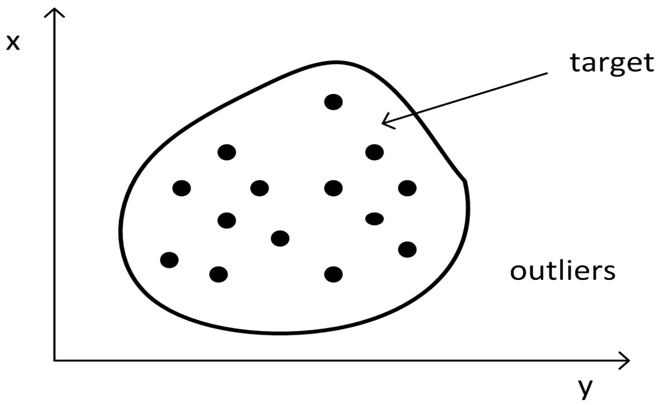


Figure 3.6: Illustration of one-class classification problem in two dimensional space.

detectors represented as circles in two dimensional space or hyper-spheres in n-dimensional space. Using Euclidean distance based matching rule means if a given data is within boundaries of a detector, then it satisfies the matching rule. The traditional definition of Euclidean distance in two dimensional space is as follows:

$$D_{Euclidean}(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \tag{3.1}$$

3.2 One-Class Classification

In one-class classification, a specified set of data is modeled in the training phase, and a new given instance is detected, if it resembles the training data. In the training phase, only one class samples are available in one-class classification, and this is the difference from conventional classification. In Tax study [56], the instances from the training class are called target instances, and the other remaining instances are called outlier instances. Figure 3.6 illustrates one-class classification problem in two dimensional space. In this figure, the black circles show the training instances, and solid line indicates a possible one-class classifier that determines the boundary between target and outlier instances.

The proposed methods to solve the one-class classification problem can be categorized into several goups.Tax [56] grouped them into three main approaches: the density estimation, the boundary methods and the reconstruction methods.

The density estimation methods estimate the density of the training data and set a threshold on this density, thus they try to find a one-class classifier. Several distributions can be assumed, such as a Gaussian or a Poisson distribution, and to test new instances, numerous tests, called discordancy tests, are used [56].

Solving a more general problem is difficult, when a limited amount of training data is used. More data might be required than the original problem to solve more general problems. This means that estimating a complete data density might also be too difficult. Therefore, only a closed boundary around the target data is optimized in the boundary methods [56]. Support Vector Machine (SVM) is

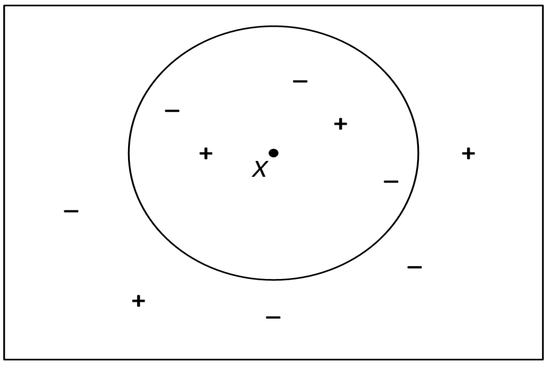


Figure 3.7: Illustration of k-NN classification.

the well-known boundary method [57].

The reconstruction methods have been constructed to model the data. A model is selected and fitted by using prior knowledge about the data and making assumptions about the generating process. A new given instance is defined in terms of a state of the generating model. If more compact representation of the target data is found, then the noise contribution is decreased. This compact representation simplifies processing without harming the information [56].

3.3 *k* Nearest Neighbor

The *k* Nearest Neighbor (*k*-NN) algorithm is the most basic instance-based method. *k*-NN classifies objects based on nearest training examples in the feature space [58]. The nearest neighbors of an instance are found using Euclidean distance. An arbitrary instance is defined by feature vector, x=x_1, ...,x_n. Then the distance between two instances x^i and x^j is calculated as follows:

$$D(x^i, x^j) = \sqrt{\sum_{r=1}^{n}(x_r^i - y_r^i)^2} \tag{3.2}$$

The given instance is classified by a majority vote of its neighbors. The number of neighbors is specified by *k* value. The neighbors are calculated from a set of instances which their correct classes are known. Figure 3.7 illustrates the *k*-NN classification of instances in two dimensional space. There are two classes, positive and negative examples presented by "+" and "-" respectively. If *k* is selected as 1, then the instance *x* is classified as positive examples. However, if *k*is chosen as 5 as in Figure 3.7, then *k*-NN classifies the instance *x* as a negative example

3.4 Local Outlier Factor

In the literature, there are too many outlier detection methods. They can be categorized as distance-based, density-based and clustering-based methods. The Local Outlier Factor (LOF), which is a density-based method, was selected to find variable self radius of each sample. LOF technique computes local outlier factor (LOF) value for each point in the dataset, indicating their degree of outlierness [59]. The LOF of a point is found based on the local density of one point's neighborhood. The point's neighborhood is defined by the *MinPts* parameter, which is the number of nearest neighbors. Points with a high LOF are treated as outliers. LOF uses the number of neighbors rather than a specific distance or similarity. In this way, this approach is able to handle data sets with varying densities [60].

Algorithm 2 presents the pseudo code of LOF implementation.

Algorithm 2: Local Outlier Factor (LOF) algorithm

```
Data: MinPts : the number of nearest neighbors
Data: input points
Result: outliers degree, LOF values, of input points
for each MinPts increasing predefined step from lower bound to upper bound do
    for each input point do
        compute k-distance value of each point
    end
    for each input point do
        calculate the local reachability density (lrd) for a point using its k nearest neighbors and
        their lrd values;
    end
    for each input point do
        compute the local outlier factor for a point using its lrd value and those of its k nearest
        neighbors, for the current value of MinPts;
    end
end
Sort local outlier factors of points in descending order.
Change LOF values of first points according to noise fraction, $\epsilon$.
```

Euclidean distance is used to find the k-nearest neighbors of each point. Then, based on the k-nearest neighbors, a local reachability density for each point is calculated given equation in Breunig et al [59]. In the next step, local outlier factor is computed using local reachability and k-nearest neighbors. All points are sorted in descending order with respect to their LOF values. Finally, LOF values of the first ϵ percentage points are changed to the biggest LOF value of the rest of the points. ϵ is a parameter and it gives opportunity to minimize effect of noisy data on other points. The runtime complexity of LOF algorithm is $O(n^2)$.

Varying density is not a problem for the LOF method, but there is an issue of how to select parameters, such as *MinPts*. Therefore it was suggested that the LOF of each point should be calculated for a range of values of *MinPts* [59]. Figure 3.8 illustrates LOF values of sample points in two-dimensional space. LOF value of a point is increased when its local density is decreased.

3.5 B-Spline

A spline is a function consisting of polynomial pieces. These polynomial pieces are joined together with certain smoothness conditions. A simple example is the polygonal function, B spline of degree 1, and its pieces are linear polynomials joined together to achieve continuity as in Figure 3.9. The points $x^i = (t_1, t_2, \ldots, t_n)$ where the function changes its character are named as knots in the theory of B splines [61]. Thus the spline function shown in Figure 3.9 has eight knots.

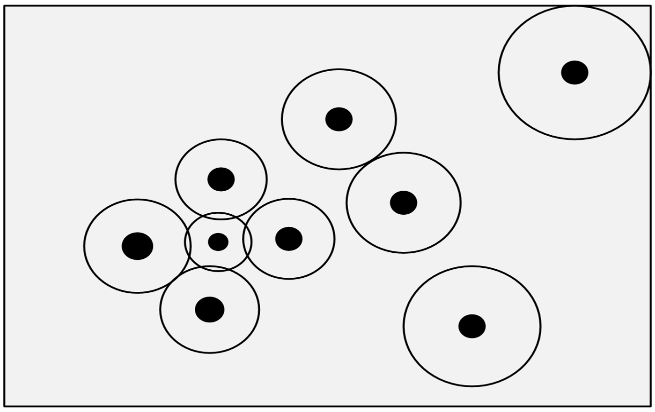


Figure 3.8: Illustration of LOF values of points.

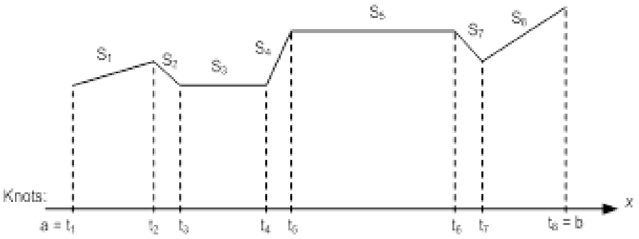


Figure 3.9: First-degree spline function.

Such a function is complicated to define in explicit terms, so it is defined as follows:

$$S(x) = \begin{cases} \mathrm{S_1(x)} & \mathrm{x} \in [\mathrm{t_1}, \mathrm{t_2}] \\ \mathrm{S_2(x)} & \mathrm{x} \in [\mathrm{t_2}, \mathrm{t_3}] \\ . & \\ . & \\ . & \\ \mathrm{S}_{n-1}\mathrm{(x)} & \mathrm{x} \in [\mathrm{t}_{n-1}, \mathrm{t}_n] \end{cases} \tag{3.3}$$

If the function *S* defined by equation (3.3) is continuous, it is called a spline of degree 1. It is characterized by the following three properties:

1. The domain of *S* is an interval [*a*, *b*].
2. *S* is continuous on [*a*, *b*].
3. There is a partitioning of the interval

$$a = t_1 < t_2 < ... < t_n = b$$

where *S* is a linear polynomial on each subinterval $[t_1, t_{i+1}]$.

In general form of B splines, a function *S(x)* is a polynomial spline of degree *n* with knots $\ldots < x_k < x_{k+1} < \ldots$ if and only if it satisfies the following two properties:

- Piecewise polynomial: *S(x)* is a polynomial of degree n within each interval $[x_k, x_k + 1)$;
- Higher-order continuity: *S(x)*, *S(1)(x)*, ... , *S(n-1)(x)* are continuous at the knots x_k.

General formula can be given as follows:

$$S(x) = \sum_{i=-\infty}^{\infty} C_i B_{i-k}^{k}(x) \tag{3.4}$$

where C_i presents coefficients and B_{i-k} shows basis functions (*k* is spline degree). If *k* is selected as 3, this is called cubic spline. The curve that is drawn using cubic spline can be seen in Figure 3.10. Figure 3.10(a) shows the drawn curve using all points, while Figure 3.10(b) shows the drawn curve using only edge points.

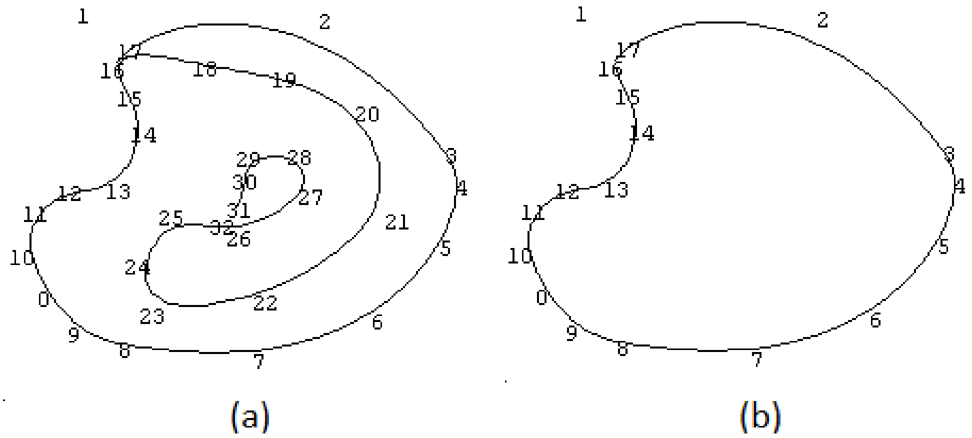


Figure 3.10: (a) Curve drawn by cubic spline with all points. (b) Curve drawn by cubic spline with edge points.

CHAPTER 4

V-SHAPED DETECTOR: A NEGATIVE SELECTION ALGORITHM

Variable self radius and detector representation has a considerable effect on the classification accuracy as explained in the previous chapter. Hence, the main concern of this study is on the definition of self space and the real-valued detector representation.

This chapter introduces the proposed NSA, variable shaped detector NSA (*V-shaped detector*). The first section describes the problem domain of the new NSA. The second section presents the new "self space" definition. In the third section, detector representation and generation process is explained in detail, and the last one gives the proposed algorithm.

4.1 Anomaly Detection

The aim of anomaly detection is to classify a given state as normal or anomalous by forming a profile of the normal state of a system. A set of features can be used to represent the state of a system based on the work by Dasgupta and Gonzalez [8].

Definition 4.1.1 *(System state space). A vector of features,* $x^i = (x^i_1, \ldots, x^i_n)$*, represents a state of the system and each feature is normalized to [0.0, 1.0]. The state space is represented by* $U \subseteq [0.0, 1.0]^n$ *for n dimensional space, where n denotes the number of features. Elements of the set are the features vectors which correspond to all possible states of the system.*

Definition 4.1.2 *(Normal subspace). A normal state of the system is represented by a set of feature vectors,* Self ⊆ U. *A complement of this set gives anomalous states of the system,* Non-Self = U - Self. *The* Self *(*Non-Self*) set can be defined by its characteristic function* x_{self}*:* $[0.0, 1.0]^n \rightarrow \{0, 1\}$.

$$x_{self} = \begin{cases} 1 \text{ if } x \in Self \\ 0 \text{ if } x \in Non\text{-}Self \end{cases} \tag{4.1}$$

Definition 4.1.3 *(Anomaly detection problem). The normal space characteristic function* x_{self} *is built using given a set of normal samples* Self' ⊆ Self. *This function should classify a given state of the system either as normal or anomalous.*

4.2 Self Space

Negative selection algorithms (NSA) generate detectors to cover the complementary space of self. Hence, the self space should be determined before the detector generation process. The self space is specified by the self set, S, but NSA can only use self samples, $S' \subseteq S$, to find the self space. Therefore, NSA model the self set, S, as a set $\hat{S}$, which is defined in terms of a set of self samples, S'. There is an assumption in this model; if an element is close enough to a self sample, then it is considered as self [27]. The closeness is defined by a radius of a self sample, r^i_{self}. Each self sample has its radius which can be different from the others. The self radius, r^i_{self}, specifies the maximum distance between an element and the self sample, s^i. The problem space is represented by $U \subseteq [0.0, 1.0]^n$ for n dimensional space. Based on these definitions, a set $\hat{S}$ can be given as follows:

$$\hat{S} = \{x \in U | \exists s^i \in S', \|s^i - x\| \leq r^i_{self}\} \tag{4.2}$$

The crucial issue for a set $\hat{S}$ is determination of self samples' radii. A self radius is calculated to consider combination of global and local features. Thus, the drawbacks of globality and locality can be minimized.

The global feature is extracted using a type of kth nearest neighbor (k-NN). First, k value is computed using *MaxCosine* value described in the next subsection. Then the average distance of kth nearest neighbor, μ_{knn}, and the standard deviation, σ_{knn}, are calculated. The sum of average k-NN distance and the standard deviation is calculated as

$$\omega_{knn} = \mu_{knn} + \sigma_{knn} \tag{4.3}$$

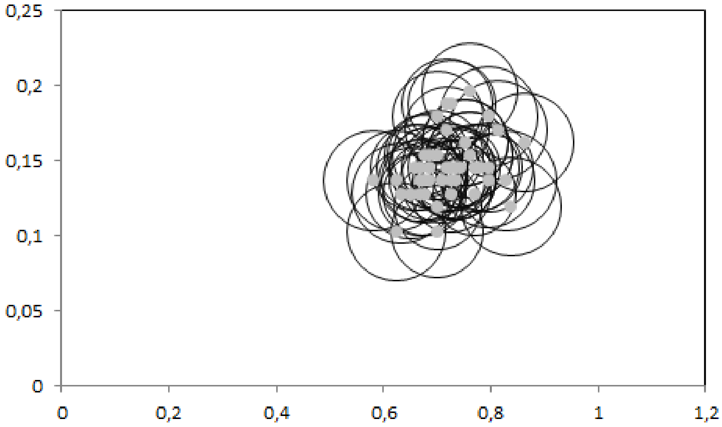
In addition to the global feature, local outlier factor, a density based outlier method, is selected to add a local feature to the self radii. In Local Outlier Factor (LOF) method, a local outlier value of each self sample which indicates its degree of outlier-ness is calculated [45]. LOF algorithm ran for the range of values of *MinPts*, from 20 to 50. The maximum of all the LOF values over this range was chosen as its LOF for each point.

$$r^i_{self} = \omega_{knn} \times LOF^i \tag{4.4}$$

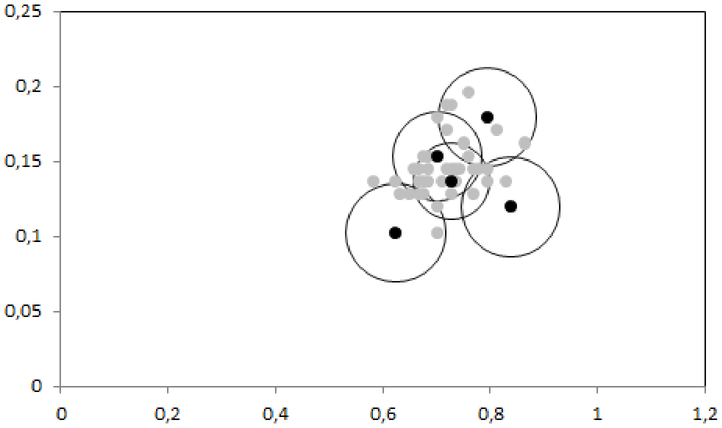
The self radii gain variability using LOF value and this provides opportunity to avoid the drawbacks of the constant self radius usage explained in the previous chapter. Figure 4.1 presents the variable self radii of setosa instances from Fisher's Iris data set. In Figure 4.1(a), the self radii of setosa instances explicitly shows the self space specified by the self model, which is defined in terms of a set of self samples. The variability of self radii can be clearly seen in Figure 4.1(b).

4.3 Detector Representation

Negative selection algorithms (NSA) generate a set of detectors that covers non-self space after the self space determination. The detectors set should be defined before the generation process. In the proposed NSA, detectors are generated as circles in two dimensional space but they are able to contain the self samples. Hence, detectors are represented generally as circles, but detectors which cover self



(a) Self space for setosa from Fisher's Iris data set.



(b) Self radii for some setosa from Fisher's Iris data set.

Figure 4.1: Self space and variable self radii for setos instances from Fisher's Iris data set.

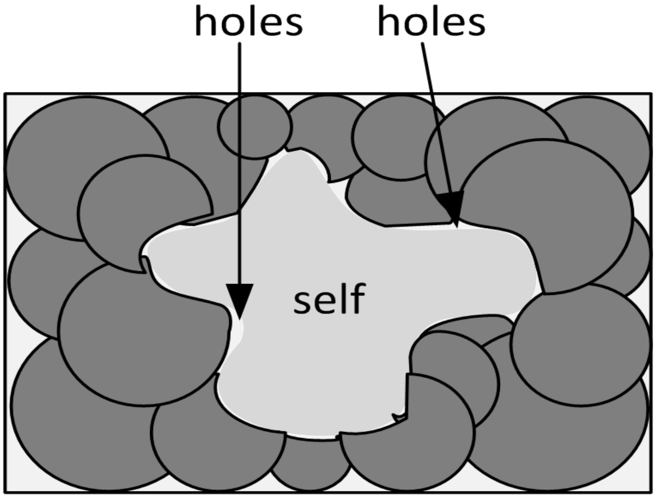


Figure 4.2: Variable shaped detectors.

samples are curved based on self samples. Thus, the area of holes between detectors and self data are minimized. Figure 4.2 represents variable shaped detectors.

In the new NSA, all self samples have to be in the same half side of a detector, vertical or horizontal. The main aim of this rule is decreasing the algorithm complexity. This representation is called variable shaped (*V-shaped detector*) detector because the shape of the half side that contains the self samples can be in any form. Figure 4.3 shows detectors that can cover self samples.

To make a curved detector, cubic b-spline interpolation method is applied. Cubic b-spline finds a function of given points and this function tries to touch all the given points. In this case, the function does not define the boundaries of the shape created by input points as described in next subsection. Therefore, to find meaningful function, edge points, $E \subseteq S'$, of the self samples should be found. In order to find edge points, edge detection algorithm was developed. Edge detection algorithm and cubic b-spline will be described in next subsections in detail.

The proposed NSA generates a detector randomly and this generated detector may cover edge points. In this case, if the covered edge points are too far from each other, the function found by cubic b-spline does not specify the real boundaries of self samples. Hence, the covered edge points should have neighborhoods that create a path between all covered points. The neighborhood is defined by self radii of the edge points. Each distance between two nearest neighbors should be less than or equal to sum of these points' self radii. In the end, cubic b-spline describes a function using the covered edge points.

The proposed NSA defines a detector, d^j, as a point, (d_1^j, d_2^j), radius, (r^j) and an array for edge points, e^j, in two-dimensional problem space. Each self sample, s^i, is described by a point, (s_1^i, s_2^i) and its self radius, r_{self}^i, on the basis of the explanations in previous subsection. In this detector representation, a detector can cover edge points of self-samples and this coverage is defined as follows:

$$IN(d^j) = \{\forall s^i \in E, \|s^i - d^j\| \leq (r_{self}^i + r^j)\} \tag{4.5}$$

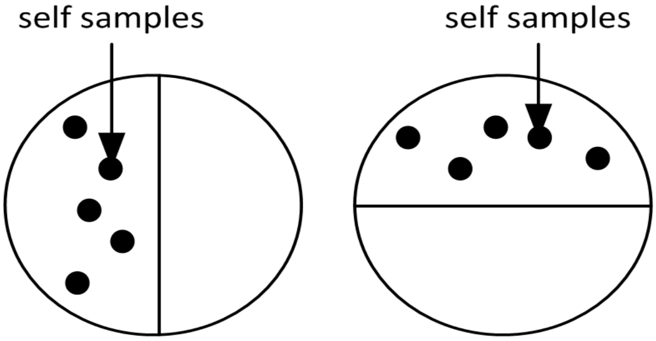


Figure 4.3: Illustration of detectors that contain the self samples.

The covered edge points are kept in the array, so the number of covered points is limited by a constant, (c).

The second rule in the detector representation mentions that all covered edge self points should be in the same half side of a detector. This rule is stated as follows:

$$HALF_{1k}(d^j) = \{\forall s^i \in IN(d^j), s^i_k \leq d^j_k\} \tag{4.6}$$

$$HALF_{2k}(d^j) = \{\forall s^i \in IN(d^j), s^i_k > d^j_k\} \tag{4.7}$$

$$HALF(d^j) = \{HALF_{11}(d^j) \vee HALF_{12}(d^j) \vee HALF_{21}(d^j) \vee HALF_{22}(d^j)\} \tag{4.8}$$

where k shows the dimension of the problem space. These equations present two halves of a detector, but there are four halves, two vertical and two horizontal, in two-dimensional space (k=2).

The other rule defines a path between all covered edge points and in this definition; a distance between the two nearest neighbors cannot be greater than the sum of these points' self radii. The path is described as follows:

$$PATH(d^j) = \{\forall s^i \in IN(d^j) | \exists s^k \in IN(d^j), s^i \neq s^k \wedge \|s^i - s^k\| \leq (r^i_{self} + r^k_{self})\} \tag{4.9}$$

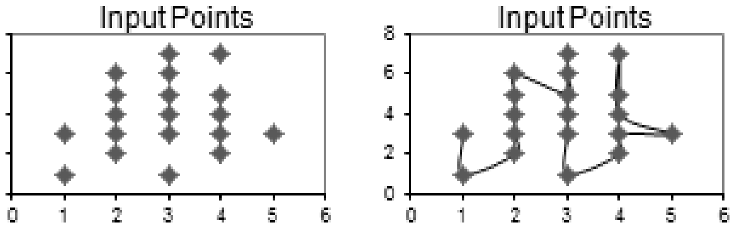
Eventually, the detectors set, D, can be defined as follows:

$$D = \{\forall d^j \in U, |IN(d^j)| = 0 \vee (|IN(d^j)| < c \wedge HALF(d^j) \wedge PATH(d^j))\} \tag{4.10}$$

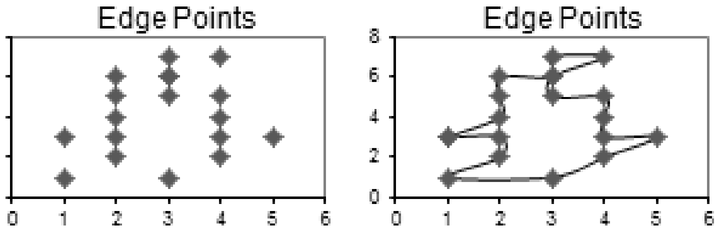
Detection of the edge points are examined in the next subsections in detail.

4.4 Edge Points Detection

The edge points need to be found in order to apply B splines interpolation. Figure 4.4 represents this fact. In the other case, B splines interpolation tries to form a function to touch all input points. The



(a) Input points and probable corresponding curve.



(b) Edge points and corresponding curve.

Figure 4.4: Input and edge points and their corresponding curves.

new method, which is called Edge Detection (ED) in Algorithm 2, was developed to find the edge points.

ED deals with each point of two-dimensional input one by one. The algorithm starts by finding the neighbors of a point and then traverses the neighbors in counterclockwise direction. Figure 4.5 shows the point, *P1*, the circle with radius, *r* and its neighbors. The point *P1* and its neighbors *P2* and *P3* form a triangle. *P2* is called current neighbor and *P3* is called next neighbor. The cosine value of the angle between the current neighbor (*P2*) and the next nearest point (*P3*) is computed based on the cosine law as follows:

$$a^2 = b^2 + c^2 - 2bccosA \tag{4.11}$$

The triangle edges a, b, c are calculated by Euclidean distance and a, b and c are represented with (x_1, y_1), (x_2, y_2) and (x_3, y_3) respectively.

$$a = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \tag{4.12}$$

$$b = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2} \tag{4.13}$$

$$c = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{4.14}$$

Thus the cosine value of angle A is

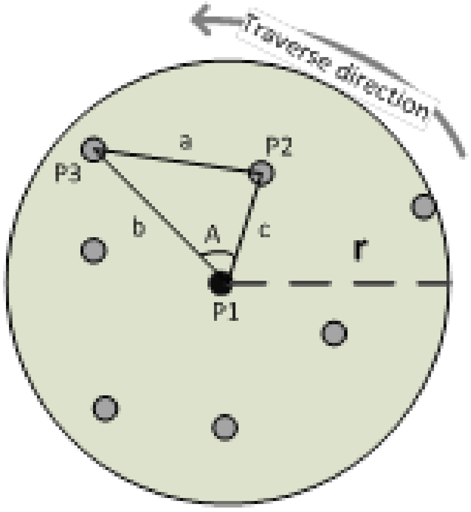


Figure 4.5: Traversing the neighbors of a point.

$$cosA = \frac{b^2 + c^2 - a^2}{2bc} \tag{4.15}$$

Afterwards the computed cosine value of the angle is compared with a predefined value, *MaxCosine*. If it is greater than the *MaxCosine*, then this point is edge point. Otherwise, algorithm makes $P3$ as current neighbor and a nearest point of $P3$ in counterclockwise direction as next neighbor. Then, *MaxCosine* is computed again for these points. Based on these explanations, the runtime complexity of ED algorithm is O($k.n$), where k shows the number of neighbors of a point and its maximum value may be the number of points, n. Therefore, ED algorithm's runtime complexity is $O(n^2)$.

The other important point is the relation between the radius of the circle and the predefined angle *MaxCosine*. In normal distribution, the inner point has at least six neighbors for 60 degree of *MaxCosine* value and four neighbors for 90 degree of *MaxCosine* value. Starting from that, the algorithm calculates the required neighbors for an inner point as follows:

$$k = \frac{360}{MaxCosine} + 1 \tag{4.16}$$

The input data sets have small real numbers, so that it is possible to make mistakes in the mentioned cosine calculation. Therefore, the division, $\frac{360}{MaxCosine}$, is increased by one in the equation (4.16) to prevent these types of errors. Once the number of the required neighbors (k) is calculated, the average distance of kth neighbors and the standard deviation from the average are computed. Finally the radius is calculated by taking the addition of the average distance and the standard deviation. The effect of the non-uniform distribution of the input data sets is minimized by adding the standard deviation to the average distance.

Based on the preceding explanations, ED algorithm is summarized as follows:

Algorithm 3: Edge points detection algorithm for two-dimensional input

```
Data: MaxCosine
Data: input points
Result: edge points
find the number of the required neighbors k = $\frac{360}{MaxCosine}$ + 1
calculate average distance between all points and their kth nearest neighbor
calculate standard deviation
assign sum of average distance and standard deviation to KNN weight
for each self point do
    multiply the KNN weight by a point's LOF and assign the result to the self radius of this point
end
for each self point do
    find all neighbors
    while traverse neighbors of a point in counterclockwise direction do
        calculate the cosine value of the angle between the current neighbor and the next nearest point based on the cosine law
        if the computed cosine value is greater than MaxCosine then
            this point is edge point
        end
        else
            if all neighbors are traversed then
                the terminal point is compared with starting points till to find a point which is not neighbor of it
                In case of finding that point, this point is inner point, otherwise it is edge point and it is kept in an edge array
            end
            else
                continue by going to the next neighbor
            end
        end
    end
end
return edge points
```

The first two weeks' data sets of 1999 DARPA were used in the experiments. The first week's data set does not include any attack, so it is accepted as training data set and the second week is used as test data set. The algorithm run for 60, 90 and 120 degrees of $MaxCosine$ value for the first week Monday of 1999 DARPA Intrusion Detection Evaluations data sets. The results of these executions are presented in Figure 4.6. Monday training data set consists of 703 input points and ED algorithm finds 467, 370 and 301 edge points for 60, 90 and 120 degrees of MaxCosine values respectively. The results show that increasing $MaxCosine$ values decrease the number of the edge points as expected.

4.5 V-Shaped Detector Algorithm

The proposed method (*V-Shaped detector*) focuses on decreasing classification errors of real valued negative selection. Therefore, variable self samples' radii and variable shaped detector representation

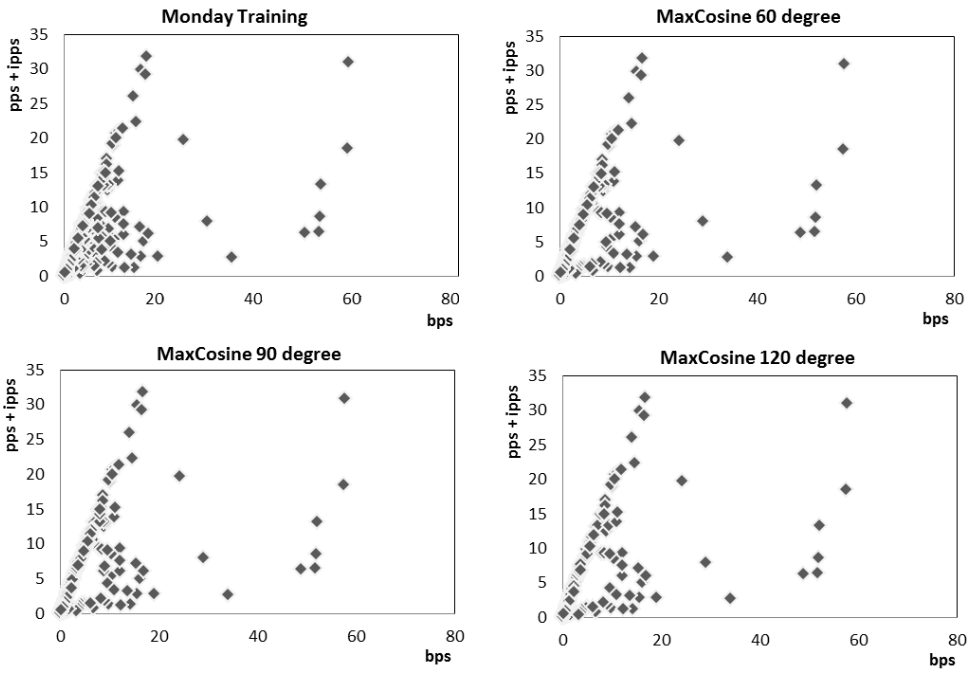


Figure 4.6: The result of Edge Detection algorithm.

are the main concerns of this study. In the new method, to get variable self samples radii, LOF and k-NN algorithms are used. In addition to that, variable sized detectors are generated as circles, and they are able to contain self-samples. Nevertheless, self samples have to be in any half side of the detector, vertical or horizontal. B-Spline interpolation method is used to find the functions of self-samples. Figure 4.7 shows the general diagram of *V-Shaped detector* architecture. There are four main processes in *V-Shaped detector*, self radii determination, detection of the edge points, detectors generation and checking the given data (test data). In the first process, the self radii of the given self samples (training data) are calculated using LOF and k-NN methods. Then, edge points of the given self samples are determined using ED algorithm. In the third process, variable sized and shaped detectors are generated considering self samples, their self radii and edge points to cover the non-self space. This process continues until the detectors' coverage reaches the given estimated detector coverage. The last process checks the given test data and classifies each instance either as normal data or as anomalous data.

The expectation from the method is to minimize classification errors, false positive and false negative.

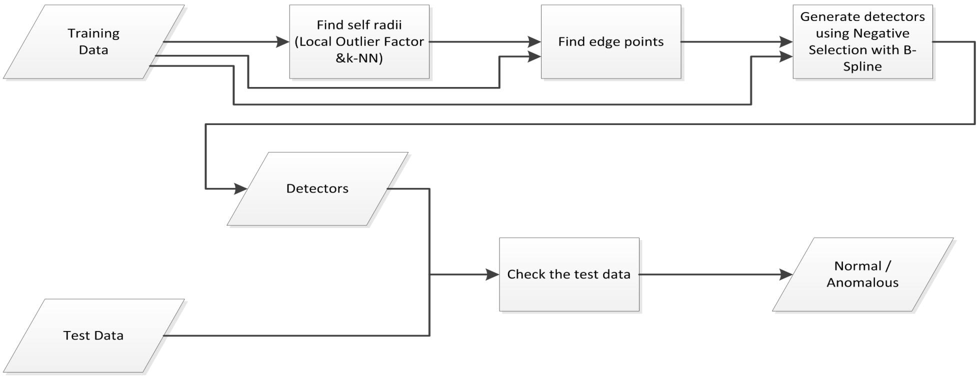

Training Data
Find self radii (Local Outlier Factor &k-NN)
Find edge points
Generate detectors using Negative Selection with B-Spline
Detectors
Check the test data
Normal / Anomalous
Test Data

Figure 4.7: General diagram of *V-Shaped detector* architecture.

The implementation of the method is presented in Algorithm 4.

Algorithm 4: Negative Selection with B-Spline (V-shaped detector)

```
Data: training dataset (self samples)
Data: test dataset
Data: estimated of non-covered non-self space, $e_{nc}$
Result: anomalous data
Calculate the volume of the self set, $V_s$
assign subtraction $V_s$ from 1 to the volume of the non-self set ($V_{ns}$)
initialize population
calculate the volume of the covered non-self, $V_{cd}$
while $V_{cd}$ is less than or equal to ($V_{ns}$ - $e_{nc}$) do
    generate new detectors using Monte Carlo algorithm
    calculate the fitness of each individual
    eliminate detectors according to the mutation ratio
    calculate the volume of the detectors, $V_{cd}$
end
generate result for test data
```

The proposed algorithm calculates self/non-self set volumes. To calculate them, some definitions about self/non-self set volumes and Monte Carlo Integration method should be described. The self/non-self space, U, is denoted by the unitary hypercube, $[0, 1]^n$. Obviously, the sum of the self space volume and the non-self space volume is equal to 1. Hence, the volume of the non-self space is defined as:

$$V_{non-self} = 1 - V_{self} \tag{4.17}$$

The entire self space is not known, so self set, S, is modeled as a set $\hat{S}$, which is defined in terms of a set of self samples. This model is described in the previous subsection. Therefore, the volume, $V_{\hat{S}}$, of the set $\hat{S}$ is used instead of the volume of the self space, V_{self}. The volume, $V_{\hat{S}}$, of the set $\hat{S}$ is calculated as:

$$V_{\hat{S}} = \int_U X_{\hat{S}}(x)\,dx \tag{4.18}$$

where the function $X_{\hat{S}}$ defines the element of the set $\hat{S}$ as follows:

$$X_{\hat{S}} = \begin{cases} 1 \text{ if } x \in \hat{S} \\ 0 \text{ if } x \notin \hat{S} \end{cases} \tag{4.19}$$

Gonzalez et al. [27] calculate the estimate of $V_{\hat{S}}$ using random sampling. In this idea, a sequence $\{x_i\}_{i=1..m}$ of random samples is generated and these random samples are uniformly distributed in U. The expected value of $X_{\hat{S}}(x_i)$, E$[X_{\hat{S}}(x_i)]$, is equal to $V_{\hat{S}}$.

$$E[X_{\hat{S}}(x_i)] = \int_U X_{\hat{S}}(x)\,dx \tag{4.20}$$

Hence, an estimate of E$[X_{\hat{S}}(x_i)]$ is also an estimate of $V_{\hat{S}}$. In addition to that, a good estimate of the mean of random variable (expected value) is the mean of a set of self samples. Therefore, the average of $\{X_{\hat{S}}(x_i)\}_{i=1..m}$ is used as an estimate $\hat{V}_{\hat{S}}$ of the self volume:

$$V_{\hat{S}} \approx \hat{V}_{\hat{S}} = \frac{\sum_{i=1}^{m} X_{\hat{S}}(x_i)}{m} \tag{4.21}$$

The estimation of the defined integral by averaging a set of random samples is known as Monte Carlo integration [27]. In the end, the algorithm is written to calculate the volume of the self set and it can be seen in Algorithm 5.

Algorithm 5: Monte Carlo algorithm to calculate the estimated self volume

```
Data: distance determines an element as self or not, selfDistance
Data: epsilonmax : maximum allowed error
Data: initial number of iterations, $m_{min}$
Data: self samples
Result: The estimated self volume, $\hat{V}_{\hat{S}}$
$maxX = inputUpperX$
$maxY = inputUpperY$
num_hits = 0
$m = 0$
repeat
    $m = m + 1$
    $x$ = random number modulo $maxX$
    $y$ = random number modulo $maxY$
    distance = Find the nearest neighbor of $(x,y)$
    if distance ≤ nearest self neighbor's radius then
        num_hits = num_hits + 1
    end
    $\hat{V}_{\hat{S}} = \frac{num\_hits}{m}$
    $epsilon = 3\sqrt{\frac{\hat{V}_{\hat{S}} - \hat{V}_{\hat{S}}^2}{m}}$
until $m \geq m_{min}$ and $epsilon \leq epsilon_{max}$;
return the estimated self volume, $\hat{V}_{\hat{S}}$
```

In the Algorithm 5, Monte Carlo Integration method is used to estimate the self volume. The algorithm randomly generates a data and it checks whether the generated data is close enough to self samples or not. If it is, hit number is increased by one. This process is repeated until a sufficient estimation precision is reached. The estimated self space is calculated by dividing the hit by the number of iteration.

The *V-shaped detector* algorithm starts with the non-self space estimation. Then, the detector population is initialized using Monte Carlo Integration. In Monte Carlo Integration algorithm, if randomly generated point is in non-self space, it is accepted as center of a detector. The radius of this detector is calculated to consider the distance of the nearest self sample. Afterwards, the generated detector checked whether it is a valid individual or not on the basis of rules defined in the previous subsection. The valid detector should have the following properties.

- The number of edge self samples in the generated detector is limited by 20.
- The positions of self-samples should be in the same half side, vertical or the horizontal, of the detector.
- There should be a path between the all edge self samples in the generated detector.

In the next step of *V-shaped detector* algorithm, to calculate the volume of the non-self space covered by detectors, the model proposed by Gonzalez et al. [27] is used again. The set should be defined before using it in this model. The covered non-self set, N_c, is defined using the detectors set, D, as follows:

$$N_c = \{x \in U | \forall s^i \in S' | \exists d^j \in D, ||s^i - x|| > r^i_{self} \wedge ||d^j - x|| \leq r^j\} \tag{4.22}$$

The volume of the covered non-self space is calculated by Monte Carlo Integration algorithm considering the covered non-self set.

In the new detector generation step, the same algorithm used in population initialization runs with the existent detector population. Then, the fitness of each valid detector, $d^j \in D$, is calculated by considering the volume of non-self space covered by only this detector. Inspired from the volume calculation of the covered non-self space, the fitness of a detector is computed as follows:

- The volume of the detector is calculated by the following formula:

$$volume(d^j) = \pi \times {r^j}^2 \tag{4.23}$$

- The volume ratio of the covered non-self space by only the detector, ${n_r}^j$, is defined as the following set definition:

$${n_r}^j = \{x \in u | \forall s^i \in S' | \forall d^k \in D, d^k \neq d^j \wedge ||s^i - x|| > r^i_{self} \wedge ||d^k - x|| > r^k\} \tag{4.24}$$

where u corresponds to the detector, d^j, space.

The fitness is defined as:

$$fitness(d^j) = volume(d^j) \times {n_r}^j \tag{4.25}$$

The volume ratio of non-self space coverage, ${n_r}^j$, is calculated by Monte Carlo Integration method as calculation of the self space volume.

Detectors elimination starts from the lowest fitness and continues till specified mutation ratio is reached. The mutation ratio should be higher than the mutation ratio of Genetic Algorithm, due to the AIS characteristic.

The new detector generation, fitness calculation and detectors' elimination steps are repeated till the volume of the covered non-self space has reached the specified coverage ratio.

The proposed algorithm consists of the three components; self radii determination, edge detection and detector generation as mentioned in the beginning of this subsection. In the first component, LOF and k-NN are used serially to find self radii, and the runtime complexities of LOF and k-NN are the same, $O(n^2)$. Therefore, the runtime complexity of the first component is $O(n^2)$. ED algorithm is used in the second component to find edge points, and its runtime complexity is $O(n^2)$. The last component is the detector generation and Monte Carlo algorithm determines this component's runtime complexity. The runtime complexity of the Monte Carlo algorithm is $O(n.p)$, where p can be greater than the n. Hence, the runtime complexity of Monte Carlo algorithm can be considered as $O(n^2)$. These three components run serially, so the higher runtime complexity specifies the proposed algorithm complexity, and all components' runtime complexity are $O(n^2)$. Therefore, the runtime complexity of the proposed algorithm is $O(n^2)$.

CHAPTER 5

EXPERIMENTATION AND RESULTS

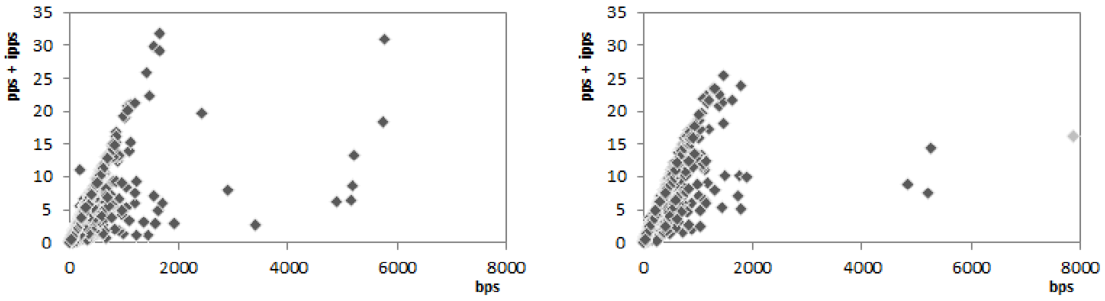
In order to find out the advantages and disadvantages of the proposed NSA, *V-shaped detector*, experiments were performed with 1999 DARPA Intrusion Detection Evaluation [62], Fisher Iris [63] and Biomedical [64] data sets. In the first part of experiments, properties of *V-detector* [65] were explored. In the other part, the experimental results were compared with those of the most used one-class method, Support Vector Machine (SVM), and the recent known NSA, *V-detector*. A well-known implementation of SVM, *LIBSVM* [66], was used to compare *V-shaped detector* with SVM. *V-detector* is a real-valued negative selection algorithm that specifies self space using boundary-aware approach and represents detectors as variable size circles. Although *V-detector* has many parameters (see Appendix A), one of the critical parameters is naive estimation option, a boolean parameter. Naive estimation option is enabled in detectors generation process, so the algorithm tries to cover non-self space with the minimum number of detector. In the SVM experiments, the parameters of *LIBSVM* were optimized (see Appendix A) using the *grid.py* script. *LIBSVM* implements some kernel function; linear, polynomial, radial basis function and sigmoid. Kernel function was selected as radial basis function (RBF), because *LIBSVM* generates more correct results with it.

The first dataset, 1999 DARPA Intrusion Detection Evaluation consists of two weeks and data sets are divided into training and test data sets according to the week. First week is attack free data set so that it was accepted as training data set. On the other hand, second week includes many attacks and it was selected as test set. Three week days (Monday, Wednesday and Friday) is selected as an input from each week. These data sets (see Appendix B) were preprocessed to be used in the experiments. Figure 5.1 represents the first and second week Monday's data sets. Figure 5.1(a) shows the first week Monday data set used in training phase because it does not include any attack. Figure 5.1(b) displays the second week Monday data set. This data set was used to test NSA because it includes an attack indicated as light grey in the Figure 5.1(b).

The other data set is Fisher Iris data set (see Appendix B), which is widely used as a benchmark of various methods. This data set was processed to reduce to two dimensions, before it was used in the experiments. Figure 5.2 shows the Fisher Iris data set after the preprocessing.

The last data set is the biomedical data set (see Appendix B), which is also used as a benchmark. This data set consists of samples, which each one includes four features. Hence, it was also processed to reduce to two dimensions, before it was used in the experiments. Figure 5.3 shows the biomedical data set after the reduction to two dimensions.

The experiment results were evaluated based on the classification accuracy which is specified by detection rate and the false alarm rate. Detection rate and false alarm rate are defined as



(a) The first week Monday.

(b) The second week Monday.

Figure 5.1: The first and second week Monday data sets.

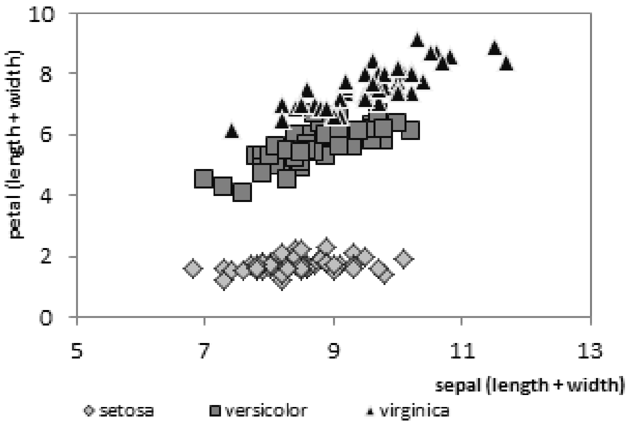


Figure 5.2: Reduced Fisher Iris Data.

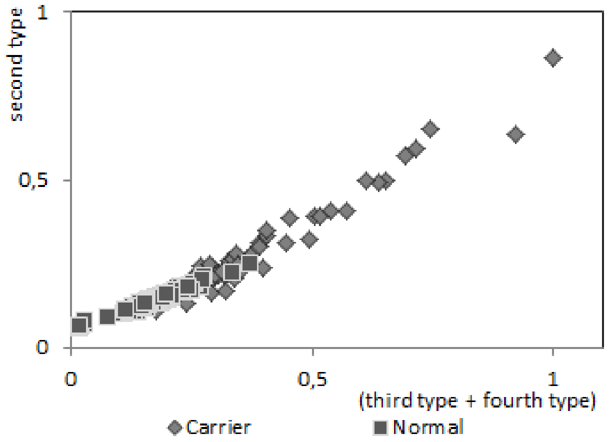


Figure 5.3: Reduced Biomedical Data.

$$DR = \frac{TP}{(TP + FN)} \tag{5.1}$$

$$FA = \frac{FP}{(FP + TN)} \tag{5.2}$$

where *TP*, *FN*, *FP*, *TN* are the counts of true positive, false negative, false positive, and true negative respectively. High detection rate and low false alarm rate is expected from a successful NSA.

Many experiments are needed in order to be able to make a comment on and an inference according to the results. Therefore, the average results of ten consequent runs are calculated and represented in the following figures and tables.

In the first experiments, the properties of *V-shaped detector* were explored and in these experiments, the proper value of *MaxCosine* was investigated considering detection rate and low false alarm rate. Figure 5.4 shows average self radii, number of edge points, detection rates and false alarm rates for different *MaxCosine* values. In these experiments, halves of versicolor and virginica samples were separately used as a training data, and all Fisher's Iris data sets as a test data.

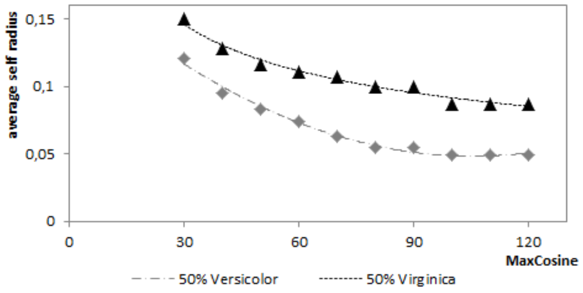
There is a relation between *MaxCosine* value and self radii, radii of self (training) samples, as mentioned in Sect. 4.5). There is direct correlation between self radii and the value of *k* as stated in equation (4.4). In addition to that, *k* value is increased while *MaxCosine* value is decreased as given in equation (4.16). The average value of self radii was noted for different *MaxCosine* values in the first experiments. Figure 5.4(a) shows the experiments results in the graph which displays the relation between *MaxCosine* and self radii explicitly. Self radii values are increased while *MaxCosine* is decreasing as expected. Beside this relation, Figure 5.4(b)presents the other relation between the number of edge points and *MaxCosine* value. *MaxCosine* value sepecifies the values of the self radii, and the self radii values affect the number of determined edge points. The number of edge points is important to specify the boundary of self space.

MaxCosine determines the self space directly through self radii. Moreover, it indirectly specifies the number of edge points, and these points define the boundary of self space. The determined self space is getting larger when *MaxCosine* is decreased. Hence, the determined self space may start to cover non-self, and this causes high false negative errors and low true positive. Therefore, detection rate is decreased. Figure 5.4(c) shows this case, namely this figure presents that the detection rate decreased while *MaxCosine* was decreasing for both of data sets.

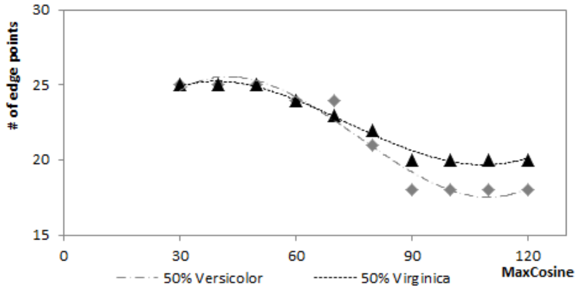
In addition to these relations, growth of the determined self space causes low false positive error. In this case, the false alarm rate is decreased as in Figure 5.4(d). The tradeoff between detection rate and false alarm rate for selecting *MaxCosine* value was inferred from Figure 5.4(c) and Figure 5.4(d). In order to highlight this tradeoff between detection rate and false alarm rate, ROC (Receiver Operating Characteristics) diagrams are used. To draw ROC diagrams, the detection and false alarm rates are needed for different values of *MaxCosine* as in Figure 5.5.

Figure 5.5 shows the ROC diagrams for different training data sets. Figure 5.5(a) presents the ROC diagram for selected 50% versicolor as training data, and Figure 5.5(b) shows the ROC diagram for selected 50% virginica as training data. Both ROC diagrams depicts that 90 degree is appropriate value for *MaxCosine*.

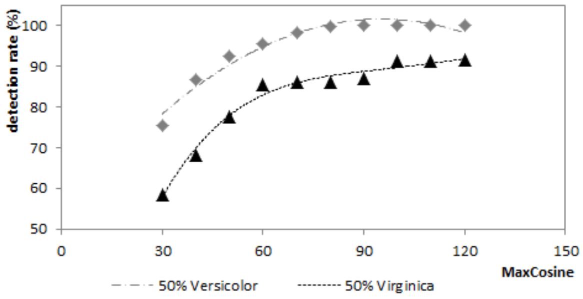
To expose strong and weak sides of *V-shaped detector*, experiments not only focused on the properties of the new method, they also carried out to compare the results of *V-shaped detector* with the results of the well-known one-class method, SVM, and another NSA, *V-detector*.



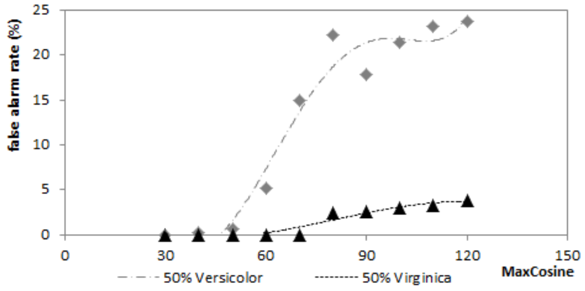
(a) *MaxCosine* versus average self radius.



(b) *MaxCosine* versus number of edge points.

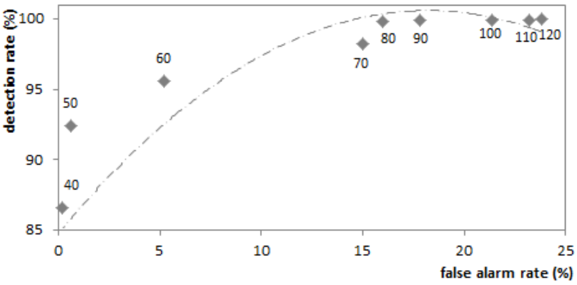


(c) *MaxCosine* versus detection rate.

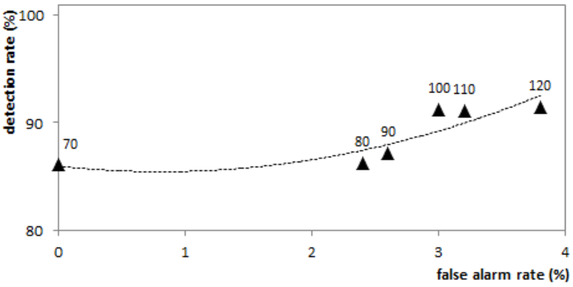


(d) *MaxCosine* versus false alarm rate.

Figure 5.4: Relations between *MaxCosine*, average self radius, detection rate and false alarm rate.



(a) ROC diagrams for 50% versicolor.



(b) ROC diagrams for 50% virginica.

Figure 5.5: ROC diagrams for Fisher's Iris data set.

5.1 Comparison with SVM

V-shaped detector and one-class SVM, *LIBSVM*, were tested using DARPA 1999 Intrusion Detection Evaluation data sets and Fischer's Iris data set. The default kernel function, Radial Basis Function (RBF), was selected in SVM. Depending on the kernel function, the parameters of *LIBSVM* were optimized (see Appendix A) in these experiments. On other side, *V-shaped detector*'s parameters, *MaxCosine* and estimated of non-covered non-self space (e_{nc}) were set to 90-120 and 0.015 (1.5%) respectively.

In the first experiments, 1999 DARPA Intrusion Detection Evaluation data sets were divided into training and test data. The first week days' data was used to train both methods, and they tested using the second week days' data. Two test cases were generated using same training data set; 100% and 50% of a training data set use to train these methods. Table 5.1 shows the experiments' results based on two criteria: detection and false alarm rates. All the results shown in Table 5.1 are average of ten consequent runs.

Table 5.1 shows that *V-shaped detector* and SVM have almost the same detection rates for 100%, 50% and 25% training data of Monday and Wednesday. The detection rate of SVM is slightly better than *V-shaped detector* for Friday data set. The false alarm rates of two methods are almost the same for 25% training data of Wednesday. On the other hand, SVM has much higher false alarm rate than *V-shaped detector* in all other test cases. Overall results in Table 5.1 presents that *V-shaped detector* has higher classification accuracy than SVM for 1999 DARPA Intrusion Detection Evaluation data sets.

In the later experiments, *V-shaped detector* and SVM were trained and tested using Fischer's Iris data. As explained in the beginning of this section, there are three types of iris, *setosa*, *versicolor*, *virginica*, and one of them is considered as normal data, while the others are considered as anomalous data. The normal data was used to train the systems and the whole data set was used to test them.

Table 5.2 shows the results of *V-shaped detector* and SVM using a Fisher's Iris dataset. Although, two methods have same detection rates for 100%, 50% and 25% of setosa data, SVM has slightly higher detection rate than *V-shaped detector* for *versicolor* and *virginica* data sets. On the other hand, *V-shaped detector* has much lower false alarm rates than SVM in all test cases. Table 5.2 presents that *V-shaped detector* makes more correct classification than SVM for Fisher's Iris dataset.

The biomedical data set is the last data set that was used to compare methods. In the biomedical data set, the blood measurements of normal patients are considered as normal data, while the blood measurements of carriers are considered as anomalous data. The systems were trained using the normal data and all samples of the data set were used to test them.

Table 5.3 shows the results of *V-shaped detector* and SVM using the biomedical dataset. The detection rates of *V-shaped detector* and *V-detector* are almost the same for 100% of training dataset. Detection rates of *V-shaped detector* are much higher than SVM for 50% and 25% of normal data. Besides this, false alarm rates of *V-shaped detector* are much lower than SVM for all test cases.

All tables show that the classification performance of *V-shaped detector* is better than SVM's classification performance explicitly. Therefore, ROC diagrams were not drawn to show comparisons between *V-shaped detector* and SVM.

Table5.1: Comparison between *V-shaped detector* and *oc*SVM using 1999 DARPA Intrusion Detection Evaluation data sets

Training data	Algorithm	Self radius / MaxCosine	DR(%)	FA(%)
Monday (100%)	SVM	RBF	100	57
	V-shaped	120	100	0.1
		90	100	0
Monday (50%)	SVM	RBF	100	67
	V-shaped	120	100	0.09
		90	100	0.03
Monday (25%)	SVM	RBF	100	62.9
	V-shaped	120	100	0.8
		90	100	0.7
Wednesday (100%)	SVM	RBF	100	64.8
	V-shaped	120	100	0.5
		90	100	0.4
Wednesday (50%)	SVM	RBF	100	53
	V-shaped	120	100	2.4
		90	100	1.5
Wednesday (25%)	SVM	RBF	100	3.2
	V-shaped	120	100	5
		90	100	5.2
Friday (100%)	SVM	RBF	100	41.2
	V-shaped	120	100	0.07
		90	86	0
Friday (50%)	SVM	RBF	100	44.5
	V-shaped	120	98	1.4
		90	98	1.6
Friday (25%)	SVM	RBF	100	83.3
	V-shaped	120	100	0
		90	86	0.1

Table5.2: Comparison between *V-shaped detector* and *oc*SVM using Fisher's Iris data set

Training data	Algorithm	Self radius / MaxCosine	DR(%)	FA(%)
Setosa (100%)	SVM	RBF	100	36
	V-shaped	120	100	0
		90	100	0
Setosa (50%)	SVM	RBF	100	56
	V-shaped	120	100	23.6
		90	100	21.2
Setosa (25%)	SVM	RBF	100	58
	V-shaped	120	99.9	1.8
		90	99.9	9.2
Versicolor (100%)	SVM	RBF	100	50
	V-shaped	120	93	0.6
		90	91.2	0.2
Versicolor (50%)	SVM	RBF	100	62
	V-shaped	120	99.8	20.6
		90	99.7	19.6
Versicolor (25%)	SVM	RBF	100	70
	V-shaped	120	95.4	15.2
		90	90.7	12.6
Virginica (100%)	SVM	RBF	100	50
	V-shaped	120	97.1	0
		90	96	2
Virginica (50%)	SVM	RBF	100	64
	V-shaped	120	92	3.8
		90	87.1	2.6
Virginica (25%)	SVM	RBF	100	72
	V-shaped	120	97.9	3.2
		90	98.8	10.8

Table5.3: Comparison between *V-shaped detector* and *oc*SVM using Biomedical data set

Training data	Algorithm	Self radius / MaxCosine	DR(%)	FA(%)
100%	SVM	RBF	50	88
	V-shaped	120	50	29.9
		90	46.8	29.6
50%	SVM	RBF	51.5	86.6
	V-shaped	120	69.5	4.2
		90	68.3	3.4
25%	SVM	RBF	41.8	88
	V-shaped	120	81.1	16.8
		90	78	14.1

5.2 Comparison with V-detector

The results generated by *V-shaped detector* and *V-detector* using DARPA 1999 Intrusion Detection Evaluation data sets and Fischer's Iris data set were compared. In these experiments, the parameters of *V-detector*, self radius (r_s), estimated coverage (c_0) and maximum number of detectors (T_{max}), were set to 0.1-0.05, 0.985 and 20 respectively. In other side, *V-shaped detector*'s parameters, *MaxCosine* and estimated of non-covered non-self space (e_{nc}) were set to 90-120 and 0.015 (1.5%) respectively.

In the first experiments, *V-shaped detector* and *V-detector* run for 1999 DARPA Intrusion Detection Evaluation data sets. Both methods were trained using the first week days' data and tested using the second week days' data. Two test cases were generated using the same training data set; 100% and 50% of a training data set were used to train these methods. Table 5.4 represents the experiments results based on three criteria: detection rate, false alarm rate and number of detectors. All the results shown in Table 5.4 are average of ten consequent runs.

In Table 5.4, for 100% training data sets, *V-shaped detector* and *V-detector* have almost the same detection rates and false alarm rates except Friday data set. The detection rate of *V-shaped detector* is better than *V-detector* for Friday data set. Two NSA have the same detection rates for 50% Monday training data, but *V-shaped detector* has better noticeable detection rate for 50% Wednesday and Friday training data. The false alarm rates of two NSAs are almost the same for 50% Friday training data, on the other hand, *V-shaped detector* has slightly better false alarm rate than *V-detector*. Both of NSAs have the same detection rate for for 25% Monday and Friday, but *V-shaped detector* has better detection rate for 25% Wednesday training data. The false alarm rates of *V-shaped detector* is slightly better. All these inferences can be seen using ROC diagrams as in Figure 5.6. Figure 5.6 shows ROC diagrams for 100%, 50% and 25% of training data. In these diagrams, self radius parameter of *V-detector* is 0.1 and *MaxCosine* parameter of *V-shaped detector* is 120 degree.

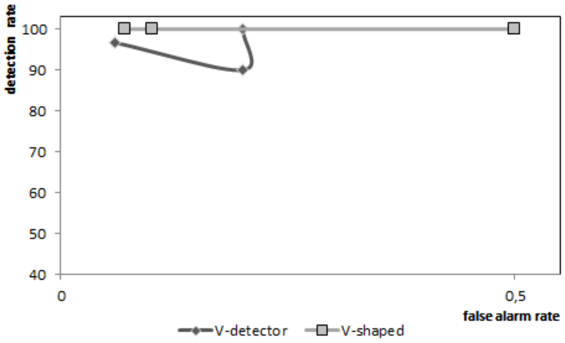
The other comparison criterion is the number of detectors, and *V-detector* needed lower number of detectors than *V-shaped detector* for all test cases of 1999 DARPA Intrusion Detection Evaluation data sets. Figure 5.7 shows this inference explicitly. In Figure 5.7, light grey bars (90 and 120 degrees) display the numbers of generated detectors by *V-shaped detector* for different *MaxCosine* values. On the other side, dark grey bars (0.1 and 0.05) show the numbers of generated detectors by *V-detector* for different self radius.

In addition to the DARPA 1999 Intrusion Detection Evaluation data sets, the two methods were also tested using Fischer's Iris data. In Fischer's Iris data set, one type of iris is considered as normal data, while the others are considered as anomalous data. The normal data was used to train the systems and the whole data set was used to test them.

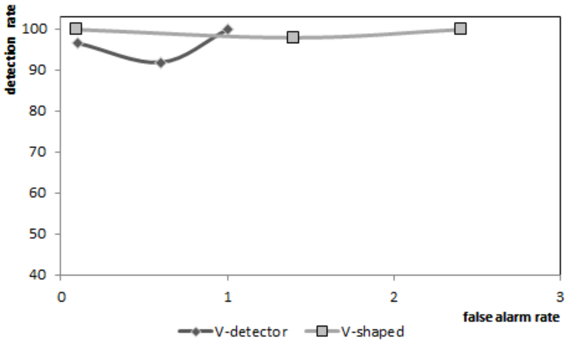
Table 5.5 illustrates the results of *V-shaped detector* and *V-detector* using the Fisher's Iris dataset. The detection rates and the false alarm rates of *V-shaped detector* and *V-detector* are almost the same for 100% of training data sets. The two NSAs have the same detection rates for 50% of setosa. Although *V-shaped detector* has slightly higher detection rate than *V-detector* for 50% of versicolor data set, the detection rate of *V-detector* is better than *V-shaped detector* detection rate for 50% virginica data sets. The two NSAs have the same false alarm rates for 50% of setosa data set, but *V-shaped detector* has better false alarm rates than *V-detector* for *versicolor* and *virginica* data sets. The detection rates of two NSAs are almost the same, but *V-shaped detector* has much lower false alarm rates than *V-detector* in all 25% of training data. ROC diagrams can show these inferences. ROC diagram is not required 100% of training data, because false alarm rates of the two methods are zero. Therefore, the methods can be compared using only detection rates. Figure 5.8 shows ROC diagrams for 50% and 25% of training

Table5.4: Comparison between *V-shaped detector* and *V-detector* using 1999 DARPA Intrusion Detection Evaluation data sets

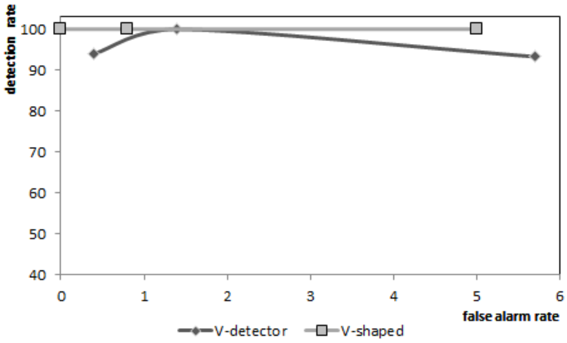
Training data	Algorithm	Self radius / MaxCosine	DR(%)	FA(%)	# of detectors
Monday (100%)	V-detector	0.1	100	0.2	6.6
		0.05	100	1.1	13.6
	V-shaped	120	100	0.1	57.8
		90	100	0	58.3
Monday (50%)	V-detector	0.1	100	1	4.6
		0.05	100	6.4	10.2
	V-shaped	120	100	0.09	51
		90	100	0.03	47.4
Monday (25%)	V-detector	0.1	100	1.4	4.5
		0.05	100	8.3	6.8
	V-shaped	120	100	0.8	59.8
		90	100	0.7	56.5
Wednesday (100%)	V-detector	0.1	96.7	0.06	2.9
		0.05	96.7	0.07	3.6
	V-shaped	120	100	0.5	10.1
		90	100	0.4	11
Wednesday (50%)	V-detector	0.1	96.7	0.1	3.2
		0.05	93.3	0.1	4.2
	V-shaped	120	100	2.4	50
		90	100	1.5	48.9
Wednesday (25%)	V-detector	0.1	93.3	5.7	1.9
		0.05	93.3	5.3	2.9
	V-shaped	120	100	5	11.8
		90	100	5.2	11.2
Friday (100%)	V-detector	0.1	90	0.2	4.7
		0.05	82	0.5	8.7
	V-shaped	120	100	0.07	34
		90	86	0	34
Friday (50%)	V-detector	0.1	92	0.6	6.5
		0.05	96	1	10
	V-shaped	120	98	1.4	61.5
		90	98	1.6	61.2
Friday (25%)	V-detector	0.1	94	0.4	4.7
		0.05	100	6.5	8.9
	V-shaped	120	100	0	40.5
		90	86	0.1	38.1



(a) ROC diagram for 100% of training data sets.



(b) ROC diagram for 50% of training data sets.



(c) ROC diagram for 25% of training data sets.

Figure 5.6: ROC diagrams for 100%, 50% and 25% of training data sets of 1999 DARPA Intrusion Detection Evaluation data sets.

Table5.5: Comparison between *V-shaped detector* and *V-detector* using Fisher's Iris data set

Training data	Algorithm	Self radius / MaxCosine	DR(%)	FA(%)	# of detectors
Setosa (100%)	V-detector	0.1	98.5	0	4.6
		0.05	98.5	0	5.9
	V-shaped	120	100	0	13.7
		90	100	0	13.2
Setosa (50%)	V-detector	0.1	100	21.2	6.3
		0.05	100	23.4	7.1
	V-shaped	120	100	23.6	7.5
		90	100	21.2	7
Setosa (25%)	V-detector	0.1	100	26.8	4.5
		0.05	100	28.4	4.7
	V-shaped	120	99.9	1.8	5.9
		90	99.9	9.2	5.1
Versicolor (100%)	V-detector	0.1	93.3	0	6.6
		0.05	93.6	0	8.1
	V-shaped	120	93	0.6	10.1
		90	91.2	0.2	11
Versicolor (50%)	V-detector	0.1	97.1	21.4	6.5
		0.05	96.2	20.2	7.4
	V-shaped	120	99.8	20.6	12.7
		90	99.7	19.6	13
Versicolor (25%)	V-detector	0.1	98.6	30.6	3.4
		0.05	98.9	32	5.2
	V-shaped	120	95.4	15.2	6.7
		90	90.7	12.6	6.7
Virginica (100%)	V-detector	0.1	96.3	0	6.4
		0.05	94.1	0	6.9
	V-shaped	120	97.1	0	14.7
		90	96	2	12.9
Virginica (50%)	V-detector	0.1	98.7	17.4	7.5
		0.05	98.3	20.4	9.9
	V-shaped	120	92	3.8	13.1
		90	87.1	2.6	10.5
Virginica (25%)	V-detector	0.1	98.7	45.4	2.8
		0.05	100	47	3.6
	V-shaped	120	97.9	3.2	9.3
		90	98.8	10.8	8.1

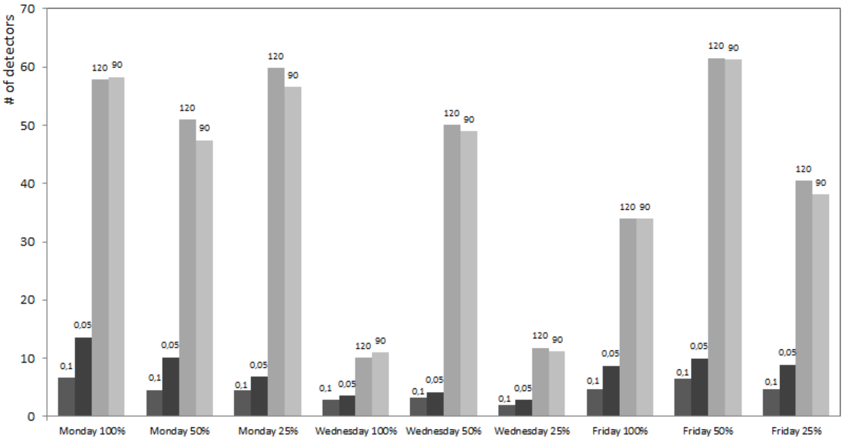


Figure 5.7: Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using 1999 DARPA Intrusion Detection Evaluation data sets.

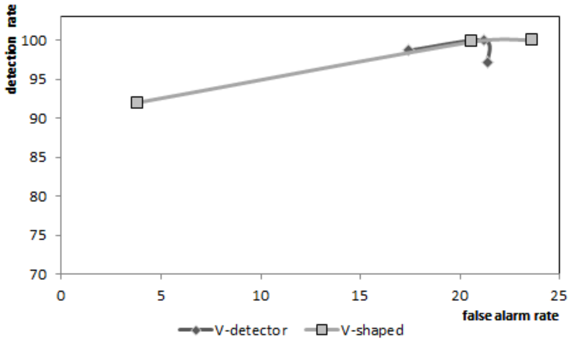
data. The self radius parameter of *V-detector* is 0.1 and *MaxCosine* parameter of *V-shaped detector* is 120 degree in these diagrams. In all test cases of Fisher's Iris data set, *V-detector* generated detectors less than *V-shaped detector*. Figure 5.9 also indicates this inference.

The last data set that was used to test methods is the biomedical data set. In the biomedical data set, the blood measurements of normal patients are considered as normal data, while the blood measurements of carriers are considered as anomalous data. The systems were trained using the normal data and all samples of the data set were used to test them.

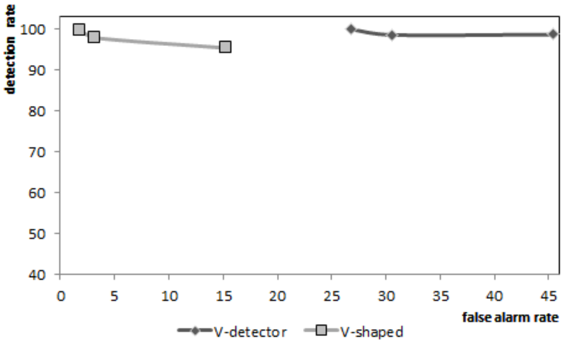
Table 5.6 shows the results of *V-shaped detector* and *V-detector* using the biomedical dataset. The detection rates and the false alarm rates of *V-shaped detector* and *V-detector* are almost the same for 100% of training dataset. Detection rates of *V-detector* are higher than *V-shaped detector* for 50% and 25% of normal data. On the other hand, false alarm rates of *V-shaped detector* are lower than *V-detector* for 50% and 25% of normal data. The last column of the table shows the number of generated detectors, and these results shows that *V-shaped detector* generates more detectors than *V-detector*. This inference is explicitly shown in Figure 5.10

Detection rates and false alarm rates in Table 5.6 were used to draw ROC diagram to be able to make comment on the comparisons between *V-shaped detector* and *V-detector*. Figure 5.11 shows this ROC diagram. Although, detection rates of *V-detector* is higher than *V-shaped detector*, *V-shaped detector* has much lower false alarm rate than *V-detector*.

The runtime complexity of *V-detector*'s detector generation part is $O(m.n)$, where $misthenumberofdetectorsandn$ is the size of training set. On the other side, the runtime complexity of *V-shaped detector*'s detector generation process is $O(n^2)$.



(a) ROC diagram for 50% of training data sets.



(b) ROC diagram for 25% of training data sets.

Figure 5.8: ROC diagrams for 50% and 25% of training data sets of Fisher's Iris data set.

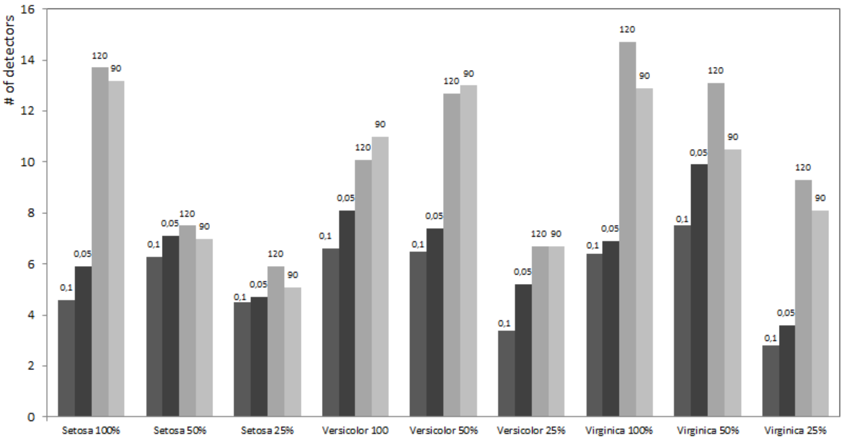


Figure 5.9: Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using Fisher's Iris data set.

Table5.6: Comparison between *V-shaped detector* and *V-detector* using biomedical data set

Training data	Algorithm	Self radius / MaxCosine	DR (%)	FA (%)	# of detectors
100%	V-detector	0.1	45	0	1.9
		0.05	49.2	0	5.9
	V-shaped	120	50	0	29.9
		90	46.8	0	29.6
50%	V-detector	0.1	83.2	14.6	4
		0.05	82.5	14.4	2.9
	V-shaped	120	69.5	4.2	17.2
		90	68.3	3.4	16.9
25%	V-detector	0.1	98.1	60.4	3.7
		0.05	98.4	60.1	4.3
	V-shaped	120	81.1	16.8	18.2
		90	78	14.1	18.4

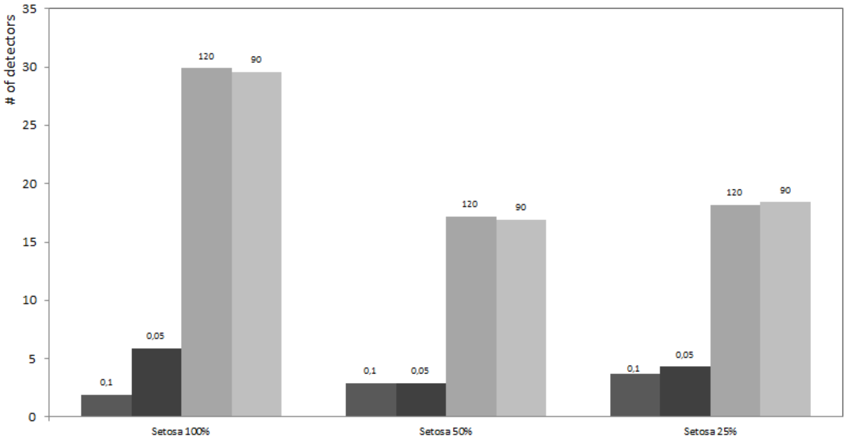


Figure 5.10: Number of generated detectors for different self radius (0.1 and 0.05) parameter of *V-detector* and different *MaxCosine* values of *V-shaped detector* using biomedical data set.

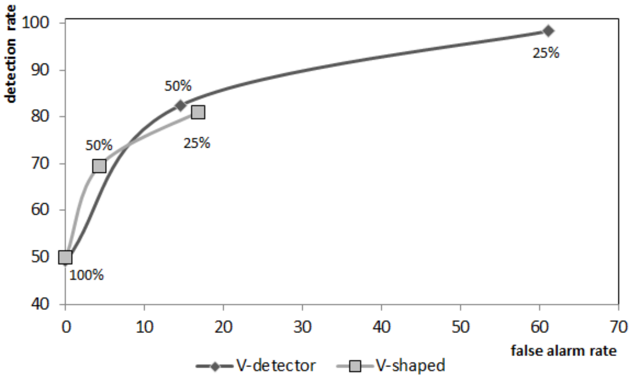


Figure 5.11: ROC diagrams for V-detector and V-shaped detector using biomedical data.

5.3 Summary

The first experiments show that the selected *MaxCosine* value affects the self radii and the number of edge points, and indirectly the determined self space through the self radii and the number of edge points. Therefore, the selected *MaxCosine* value has an enormous effect on the classification accuracy of *V-shaped detector*. To find the appropriate value of *MaxCosine*, the tradeoff between detection rate and false alarm rate should be considered.

V-shaped detector classifies the given data into one of the two classes after the training phase, so it is proper for one-class classification problem. Hence, *V-shaped detector* was compared with the well-known one-class method, SVM. Evaluation of the experiments based on the detection and false alarm rates denotes that *V-shaped detector* has higher classification accuracy than SVM. *V-shaped detector* fives satisfactory solution for one-class classification problem.

V-shaped detector is a new NSA, so it was compared with other well-known NSA, *V-detector*. Although experiment results of *V-shaped detector* and *V-detector* were very close to each other, *V-shaped detector* is slightly better than *V-detector*, when a part of normal samples were used as training data. On the other hand, *V-detector* generates fewer detectors than *V-shaped detector*, because it focuses on the optimization of detectors distribution, while *V-shaped detector* does not take into account that optimization.

Overall experiment cases show that *V-shaped detector* generates comparable results when training data compose of all the normal samples. They also indicate that *V-shaped detector* generates better results when training data include the normal samples partially.

CHAPTER 6

CONCLUSION

In machine learning, there are many learning methods that are inspired by the biological mechanisms. Genetic algorithm and neural network are the well-known biologically inspired computational models. Genetic algorithm mimics the principles and processes of natural evaluation. On other hand, neural network is inspired by the network or circuit of biological neurons and mimics the properties of biological neurons. Biological immune system is the other biological system that has various computational mechanisms: pattern recognition, memory, distributed processing, self organizing, etc. Inspired by the principles and processes of the biological immune system, many computational intelligent models were developed, and this type of models is called Artificial Immune Systems (AIS). These AIS models are categorized mainly into immune network model, clonal selection, negative selection and danger theory.

Negative selection algorithm is one of the earliest AIS methods. It is inspired by T-cells generation and maturation process. This study aimed to the state-of-art techniques in negative selection algorithms (NSA) and tried to define critical issues about them. The determined major issues are the self space determination and detectors coverage from NSA classification perspective.

This thesis study introduces a novel negative selection algorithm called *V-shaped detector*. The proposed NSA is real-valued NSA and this gives the opportunity to use other machine learning and mathematical methods. In this manner, the existent machine learning and mathematical methods are modified and used to obtain more correct self space determination and detectors' coverage. In this way, this approach facilitates the increasing classification accuracy of NSA.

V-shaped detector uses the *k*-nearest neighbor and local outlier factor to determine self space. These two machine learning methods are combined to elicit different features of the given data, so it becomes possible to find more correct self space. Beside this, *V-shaped detector* generates variable shaped detectors using cubic spline to maximize non-self coverage. This numerical analysis method, cubic spline, cannot be directly applied. If it is applied, the meaningful results cannot be obtained. Hence, the given training data is processed to find edge points which reveal the shape of the self space. Then, cubic spline is applied to the edge points within a detector region to obtain variable shaped detector. Thus, these methods enhance NS algorithm as follows:

- The use of cubic spline makes it possible to form variable shaped detectors. Variable shaped representation gives detectors more adaptation ability. Thus, it provides maximum coverage of non-self space.
- Local Outlier Factor and k-NN make this method to adapt to self space. In this manner, it finds more correct boundary for self space.

V-shaped detector is compared with the well-known one class method, SVM, and NSA, *V-detector*. All of them were tested using different data sets in the experiments. The overall classification accuracy of *V-shaped detector* is much better than SVM. Significant decrement occurred in SVM classification accuracy, when SVM was trained with the part of training data. On the other hand, *V-shaped detector* classification accuracy decreased much less than SVM. This event shows that *V-shaped detector* is more capable than SVM to adapt the given data.

In the other comparison, the overall classification accuracy of *V-shaped detector* is slightly better than *V-detector*, when all of the self samples are used as training data. Besides, *V-shaped detector* has better classification accuracy than *V-detector*, when the partial of self samples is used as training data. The experimental results reveal that *V-shaped detector* is an appropriate method to form a normal profile of a system and to solve anomaly detection problems.

In this work, optimization of the detectors' distribution was not taken into account. On the other hand, *V-detector* uses some statistical analysis method to optimize detectors distribution. Therefore, *V-shaped detector* generates more detectors than *V-detector* to cover non-self space. This means that it requires more space to keep the generated detectors. In the future work, studies will probably focus on how to decrease the number of detectors with similar classification accuracy.

REFERENCES

[1] E. Hart and J. Timmis. Application areas of ais: The past, the present and the future. volume 8, pages 191–201. Applied Soft Computing, 2008.

[2] S. Forrest, A. Perelson, L. Allen, and R. Cherukuri. Self-nonself discrimination in a computer. pages 202–212. In Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, 1994.

[3] S. A. Hofmeyr and S. Forrest. Architecture for an artificial immune system. volume 8, pages 443–473. Evolutionary Computation Journal, 2000.

[4] J. W. Haines, L. M. Rossey, and R. P. Lippmann. Extending the darpa off-line intrusion detection evaluations. 2001.

[5] A. A. Freitas and J. Timmis. Revisiting the foundations of artificial immune systems for data mining. volume 11, pages 521–540. IEEE Transactions on Evolutionary Computation, 2007.

[6] S. M. Garrett. How do we evaluate artificial immune systems? volume 13, pages 145–178. Evolutionary Computation, 2005.

[7] Al-Enezi, J. Abbod, and M. Alsharhan. Artifical immune systems - models, algorithms and application. pages 118–131. International Journal of Research and Reviews in Applied Sciences, 2010.

[8] D. Dasgupta and F. Gonzalez. An immunity-based technique to characterize intrusions in computer networks. pages 1081–1088. IEEE Transactions on Evolutionary Computation, 2002.

[9] Z. Ataser and F.N. Alpaslan. Self-adaptive negative selection using local outlier factor. pages 161–169. Computer and Information Sciences III (27th International Symposium on Computer and Information Sciences), 2013.

[10] L. N. de Castro and F. J. Von Zuben. Learning and optimization using the clonal selection principle. volume 6, pages 239–251. IEEE Transactions on Evolutionary Computation, 2002.

[11] N. Jerne. Towards a network theory of the immune system. volume 125, pages 373–389. Annals of Immunology, 1974.

[12] P. Matzinger. The danger model: A renewed sense of self. volume 296, pages 301–305. Science, 2002.

[13] J. Timmis, M. Neal, and J. Hunt. An artificial immune system for data analysis. volume 55, pages 143–150. Biosystems, 2000.

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

[15] L. N. de Castro and F. J. V. Zuben. ainet: An artificial immune network for data analysis. volume 14, pages 231–259. in Data Mining: A Heuristic Approach - Idea Group Publishing, 2001.

[16] L. Liu and W. Xu. A cooperative artificial immune network with particle swarm behavior for multimodal function optimization. pages 1550–1555. IEEE Congress on Evolutionary Computation, 2008.

[17] G. P. Coelho and F. J. Von Zuben. A concentration-based artificialimmune network for continuous optimization. pages 108–115. IEEE Congress on Evolutionary Computation, 2010.

[18] G. P. Coelho and F. J. Von Zuben. A concentration-based artificial immune network for multi-objective optimization. pages 343–357. International Conference on Evolutionary Multi-Criterion Optimization, 2011.

[19] Y. Zhong and L. Zhang. An adaptive artificial immune network for supervised classification of multi-/hyperspectral remote sensing imagery. volume 50, pages 894–909. IEEE Transactions on Geoscience and Remote Sensing, 2012.

[20] L. N. de Castro and F. J. V. Zuben. Artificial immune systems: part i - basic theory and applications. Technical Report DCA-RT 01/99 - School of Computing and Electrical Engineering, State University of Campinas, 1999.

[21] L. N. de Castro and F. J. V. Zuben. The clonal selection algorithm with engineering applications. pages 36–37. Workshop on Artificial Immune Systems and Their Applications, 2000.

[22] J. Brownlee. Clonal selection theory and clonalg - the clonal selection classification algorithm (csca). Centre for Intelligent Systems and Complex Processes (CISCP), Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology, Tech. Rep. 2-02, 2005.

[23] D. A. C. Barone L. O. V. B. Oliveira, R. L. M. Motay. Clonal selection classifier with data reduction: Classification as an optimization task. IEEE World Congress on Computational Intelligence, 2012.

[24] J. Li, H. Gao, and S. Wang. A novel clone selection algorithm with reconfigurable search space ability and its application. volume 6, pages 612–616. Fourth International Conference on Natural Computation, 2008.

[25] F. Gonzalez, D. Dasgupta, and J. Gomez. The effect of binary matching rules in negative selection. pages 195–206. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2003), 2003.

[26] F. Gonzalez, D. Dasgupta, and R. Kozma. Combining negative selection and classification techniques for anomaly detection. volume 1, pages 705–710. Proceedings of the 2002 Congress on Evolutionary Computation, 2002.

[27] F. Gonzalez, D. Dasgupta, and L. F. Nino. A randomized real-valued negative selection algorithm. pages 261–272, 2003.

[28] S. T. Powers and J. He. Evolving discrete-valued anomaly detectors for a network intrusion detection system using negative selection. pages 41–48. In The 6th Annual Workshop on Computational Intelligence (UKCI '06), 2006.

[29] J. Kim and P. J. Bentley. An evaluation of negative selection in an artificial immune system for network intrusion detection. pages 1330–1337. In Proceedings of GECCO, 2001.

[30] S. A. Hofmeyr and S. Forrest. Immunity by design: An artificial immune system. volume 2, pages 1289–1296. In Proceedings of the Genetic and Evolutionary Computation Conference, 1999.

[31] J. Kim and P. Bentley. An artificial immune model for network intrusion detection. In: 7th European Congress on Intelligent Techniques and Soft Computing, 1999.

[32] U. Aickelin and S. Cayzer. The danger theory and its application to artificial immune systems. pages 141–148. Proceedings of the 1st Internat Conference on ARtificial Immune Systems, 2002.

[33] J. Greensmith, U. Aickelin, and J. Twycross. Detecting danger: Applying a novel immunological concept to intrusion detection systems. Proceedings of the 6th International Conference in Adaptive Computing in Design and Manufacture, 2004.

[34] J. Kim, J. Greensmith, J. Twycross, and U. Aickelin. Malicious code execution detection and response immune system inpired by the danger theory. Proceedings of the Adaptive and Resilient Computing Security Workshop (ARCS-05), 2005.

[35] M. Roper. Artifical immune systems, danger theory, and the oracle problem. pages 125–126. Testing: Academic and Industrial Conference - Practice and Research Techniques, 2009.

[36] U. Aickelin and J. Greensmith. Sensing danger: Innate immunology for intrusion detection. volume 12, pages 218–227. Information Security Technical Report, 2007.

[37] Y. Zhu and Y. Tan. A danger theory inspired learning model and its application to spam detection. pages 382–389. Proceedings of the Second international conference on Advances in swarm intelligence, 2011.

[38] F. Gonzalez and D. Dasgupta. Anomaly detection using real-valued negative selection. volume 4, pages 383–403. Genetic Programming and Evolvable Machines, 2003.

[39] Z. Ji and D. Dasgupta. V-detector: An efficient negative selection algorithm with "probably adequate" detector coverage. volume 179, pages 1390–1406. Information Sciences, 2009.

[40] J. Zeng, X. Liu, T. Li, C. Liu, L. Peng, and F. Sun. A self-adaptive negative selection algorithm used for anomaly detection. volume 19, pages 261–266. Progress in Natural Science, 2009.

[41] S. Balachandran, D. Dasgupta, F. Nino, and D. Garrett. A framework for evolving multi-shaped detectors in negative selection. pages 401–408. In Proceedings of the 2007 IEEE symposium on foundations of computational intelligence, 2007.

[42] L. Xii X. Yuel, F. Zhang and D. Wangl. Optimization of self set and detector generation base on real-value negative selection algorithm. 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering, 2010.

[43] G. B. Bezerra, T. V. Barra, L. Nunes de Castro, and F. J. Von Zuben. Adaptive radius immune algorithm for data clustering. pages 290–30. In: Artificial Immune Systems: 4th International Conference, 2005.

[44] F. Gonzalez and D. Dasgupta. An immunogenetic technique to detect anomalies in network traffic. pages 1081–1088. In Proceedings of the genetic and evolutionary compuation conference, 2002.

[45] Z. Ji and D. Dasgupta. Applicability issues of the real-valued negative selection algorithms. pages 111–118. In Proceedings of the genetic and evolutionary computation conference, 2006.

[46] Z. Ji and D. Dasgupta. A boundary-aware negative selection algorithm. In Proceedings of the international conference on artificial intelligence and soft computing, 2005.

[47] Z. Ji and D. Dasgupta. Real-valued negative selection algorithm with variable-sized detectors. In proceeding of: Genetic and Evolutionary Computation, 2004.

[48] Z. Ji and D. Dasgupta. Revisiting negative selection algorithms. volume 15, pages 223–251. MIT Evolutionary Computation, 2007.

[49] D. Dasgupta and F. Nino. A comparison of negative and positive selection algorithms in novel pattern detection. volume 1, pages 125–130. IEEE International Conference on Systems, Man, and Cybernetics, 2000.

[50] T. Stibor, P. Mohr, J. Timmis, and C. Eckert. Is negative selection appropriate for anomaly detection? pages 321–328. Proceedings of the 2005 conference on Genetic and evolutionary computation, 2005.

[51] T. Stibor, J. Timmis, and C. Eckert. A comparative study of real-valued negative selection to statistical anomaly detection techniques. pages 262–275. Proceedings of the 4th international conference on Artificial Immune Systems, 2005.

[52] J. Chen, F. Liang, and D. Yang. Dynamic negative selection algorithm based on match range model. pages 847–851. Proceedings of the 18th Australian Joint conference on Advances in Artificial Intelligence, 2005.

[53] M. Middlemiss. Positive and negative selection in a multilayer artificial immune system. The Information Science Discussion Paper Series, 2006.

[54] V. D. Kotov and V. I. Vasilyev. Artificial immune systems based intrusion detection system. pages 207–212. Proceedings of the 2nd international conference on Security of information and networks, 2009.

[55] Z. Ji. Negative selection algorithms: From the thymus to v-detector. A Dissertation for the Doctor of Philosophy, The University of Memphis, 2006.

[56] D. M. J. Tax. One-class classification. A Dissertation for the Doctor of Philosophy, Delft University of Technology, 2001.

[57] O Mazhelis. One-class classifiers: A review and analysis of suitability in the context of mobile-masquerader detection. volume 36, pages 29–48. South African Computer Journal, 2006.

[58] T. M. Mitchell. Machine learning. McGraw Hill, 1997.

[59] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. pages 759–771, 2000.

[60] H. Xiong, G. Pandey, M. Steinbach, and V. Kumar. Enhancing data analysis with noise removal. volume 18, pages 304–319, 2006.

[61] W. Cheney and D. Kincaid. Numerical mathematics and computing. Brooks/Cole, 1985.

[62] MIT Lincoln Labs. Darpa intrusion detection evaluation. 1999. `http://www.ll.mit.edu/IST/ideval/index.html`.

[63] R.A. Fisher. Fisher's iris datafile. `http://lib.stat.cmu.edu/DASL/Datafiles/Fisher'sIris.html`.

[64] Statlib: Biomedical dataset. `http://lib.stat.cmu.edu/`.

[65] Z. Ji. V-detector implementation. Software available at `http://www.zhouji.net/prof/vdetector.html`.

[66] C. C. Chang and C. J. Lin. LIBSVM: A library for support vector machines. 2:1–27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[67] C. Cortes and V. Vapnik. Support-vector networks. volume 20, pages 273–297. Machine Learning, 1995.

[68] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. *Department of Computer Science National Taiwan University*, 2010.

APPENDIX A

TEST METHODS AND PARAMETERS

V-shaped detector classifies the given data either as normal or as anomalous, and this is the one-class classification problem. Therefore, it was compared with the well-known one class classification method, Support Vector Machine (SVM). In the experiments, well-known implementation of SVM, LIBSVM [66], was used. *V-shaped detector* is a new NSA, so it was also compared with the well-known other NSA, *V-detector* [65]. The next sections give information about methods and parameters' values of their implementation.

A.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is probably the most used method for one-class classification problems. We briefly introduce the basic explanations about SVM, because there are significant studies about the underlying theory behind SVM [67]. In this classification task, data is separated into training and test data sets. In the training data set, each instance consists of a "target value" (class label) and several attributes, i.e. features or observed variables. The aim of SVM models the training data and specifies "the target value" of the given test data [68]. SVM has the advantage of clear connection to the underlying statistical learning theory, but it does not generate reasonable results without experience to choose proper control parameters. Kernel function selection for a problem domain is the biggest limitation of SVM.

Training and test data must be normalized to range [0, 1] and formatted before using them in SVM experiments. Hence, training and test data were reformatted for SVM as follows:

Target value (1 or -1) feature-id2: feature-value feature-id2: feature-value

Parameters of LIBSVM are given as follows:

- svm_type : set type of SVM (default 0)
  0. C-SVC (multi-class classification)
  1. nu-SVC (multi-class classification)
  2. one-class SVM
  3. epsilon-SVR (regression)
  4. nu-SVR (regression)
- kernel_type : set type of kernel function (default 2)

0. linear: u’*v
1. polynomial: $(gamma * u' * v + coef0)^{degree}$
2. radial basis function: $exp(-gamma * |u - v|^2)$
3. sigmoid: tanh(gamma*u’*v + coef0)
4. precomputed kernel (kernel values in training_set_file)

- degree : set degree in kernel function (default 3)
- gamma : set gamma in kernel function (default 1/num_features)
- coef0 : set coef0 in kernel function (default 0)
- cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
- nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
- epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)
- cachesize : set cache memory size in MB (default 100)
- epsilon : set tolerance of termination criterion (default 0.001)
- shrinking : whether to use the shrinking heuristics, 0 or 1 (default 1)
- stimates, 0 or 1 (default 0)
- weight : set the parameter C of class i to weight*C, for C-SVC (default 1)
- n-fold cross validation mode
- quiet mode (no outputs)

SVM was used for one-class classification problems in the experiments, so *svm_type* parameter was chosen as *one-class SVM*. In the experiments, kernel functions were also explored, and radial basis function was selected as kernel function based on the SVM classification accuracy. The other parameters (C and gamma) fort the corresponding kernel function were optimized using *grid.py* script as follows:

```
$ python grid.py -v 5 iris-100setosa_train.txt

...

0.03125 0.0078125 100.0
```

This output means that best *C* and *gamma* are equal to 0.03125 and 0.0078125 respectively with five-fold 100% cross-validation rate.

Later, LIBSVM is trained with the optimized parameters as follows:

```
$ svm-train -s 2 -c 0.03125 -g 0.0078125 iris-100setosa_train.txt iris_model

*

optimization finished, #iter = 6
```

```
obj = 312.473132, rho = 24.998428

nSV = 25, nBSV = 25
```

In the last, LIBSVM is tested using test dataset.

```
$  svm-predict iris-100setosa_test.txt iris_model iris.out

Accuracy = 88% (132/150) (classification)
```

A.2 Variable-Sized Detectors Negative Selection Algorithm (V-detector)

Ji and Dasgupta [47] proposed a real-valued negative selection algorithm with variable-sized detectors (*V-detector*). The algorithm randomly generates a center of a detector (circle) which must not be within the circle of a self sample. A predefined self radius is used for all self samples. The radius of the generated detector is dynamically adjusted until the boundary of the detector region comes in contact with the center of a self sample. This method is called as boundary-aware. The algorithm is terminated, when the detector covers the predefined proportion of non-self space, or a predefined number of detectors are generated.

V-detector [65] requires normalized data to run. Therefore, training and test data were normalized to range [0, 1] before using them in the experiments. In the experiments, naive estimation option of *V-detector* was enabled to get optimized detectors distribution. Beside this, we ran *V-detector* algorithm using the following parameters.

- estimated non-self space coverage ($c0$): 98.5%
- self radius: coverage (r_s): Two values were used (0.1 and 0.05)
- maximum number of fetectors (T_{max}): 20

APPENDIX B

TEST DATA SETS AND PREPROCESSING

B.1 1999 DARPA Intrusion Detection Evaluation Data Sets

The DARPA 1999 Intrusion Detection Evaluations consisted of comprehensive technical evaluations of research intrusion detection systems.

Figure B.1 shows the isolated 1999 DARPA test bed network. In the test bed, background traffic which was initiated by hundreds of users, sourced from tens of hosts and destined to thousands of hosts, was simulated. The left hand side of Figure B.1 shows the inside of the fictional Eyrie Air Force base and the right hand side shows the Internet. Inside victim machines (SunOS, Solaris, Linux, and Windows NT) and the router were exposed to automated attacks. More than 200 instances of 58 different attacks were implemented, and related data about attacks were collected. Machines labeled as "sniffer" in Figure B.1, capture packets transmitted over the attached network segment using tcpdump[4].

Inspired by the similar methods, only outside *tcpdump* data was used as an input. This data contains the contents of every packet transmitted between computers inside and outside of a simulated military base. Outside *tcpdump* data was collected for a month by a *tcpdump* packet sniffer, but only first two weeks' data was examined. First week's data, which is attack free, is selected as training data, and second week's data, which include some attacks, is used as test data.

Outside tcpdump data of second week was filtered for *marx* machine, so that *marx* machine is the source or the destination of each network packet. Therefore, the network attacks which source or

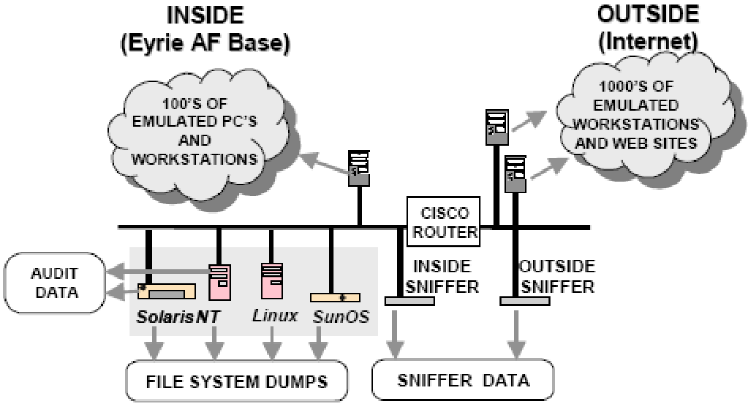


Figure B.1: Block diagram of 1999 DARPA test bed [4].

TableB.1: Second week attacks description.

Day	Attack Name	Attack Type	Start	Duration
Mon	Back	DOS	09:39:16	00:59
Wed	Satan	PROBE	12:02:13	02:29
Fri	Neptune	DOS	11:20:15	04:00

destination of them is *marx* machine are presented in Table B.1.

Outside *tcpdump* data was filtered for a specific computer (hostname *marx* - IP: 172.16.114.50) using the following command. This command filters outside *tcpdump* data which was collected on Monday.

```
tcpdump -r Monday_outside.tcpdump -w Monday_filter.tcpdump host 172.16.114.50
```

After filtering the data, *tcpstat* tool was used to get traffic statistics. Three parameters, number of bytes per second (*bps*), number of packets per second (*pps*) and number of *ICMP* packets per second (*ipps*), were selected to detect attacks. These parameters were sampled each minute using *tcpstat*. The following command gathers traffic statistics.

```
tcpstat -r Monday_filter.tcpdump -o "%B %p %C
n" 60 <Monday_tcpstat_60s_99.output
```

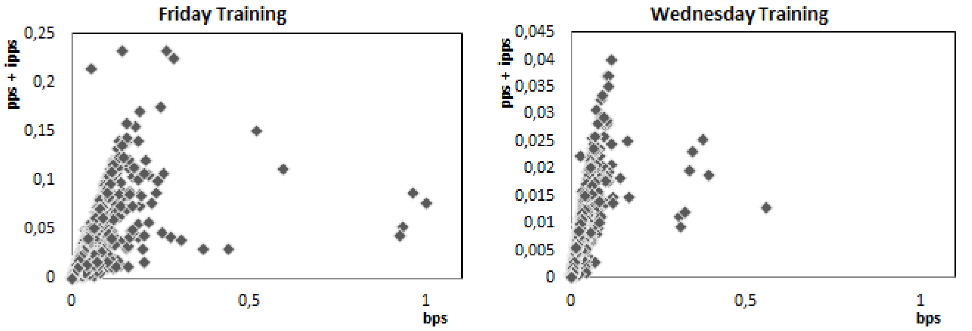
These three attributes are reduced to two dimensional inputs by taking number of bytes per second and the sum of number of packets per second (*pps*) and number of *ICMP* packets per second. Figure B.2 represents the output of *tcpstat* for the first week's outside *tcpdump* data. The graph was drawn after the outputs of *tcpstat* was reduced to two dimensional data set and then, normalized to range [0, 1].

B.2 Fisher's Iris Data Set

The other data set is Fisher Iris Data and this dataset consists of 50 samples from each of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample; they are the length and the width of sepal and petal. This data set is reduced to two dimensions by taking sum of length and width of sepal and petal. Figure B.3 shows the reduced Fisher Iris Data Set.

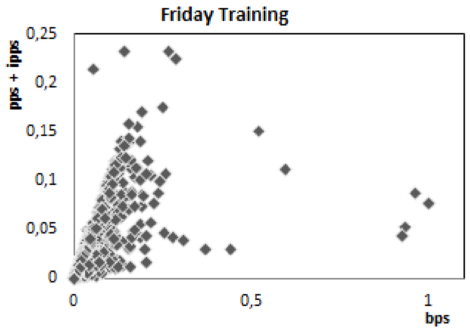
B.3 Biomedical Data Set

The third dataset, biomedical dataset, includes blood measurement which was used to screen a genetic disorder. The blood measurements of 127 normal patients were selected as training data. All the blood measurements of 127 normal patients and 75 carriers were used as test data. Each patient has four types of blood measurement but in experiments, training and test data consists of the second type and the sum of the third and the fourth types. Figure B.4 shows the reduced Biomedical Data Set



(a) Output of tcpstat for Monday's outside tcpdump.

(b) Output of tcpstat for Wednesday's outside tcpdump.



(c) Output of tcpstat for Friday's outside tcpdump.

Figure B.2: Output of tcpstat for the first week's outside tcpdump.

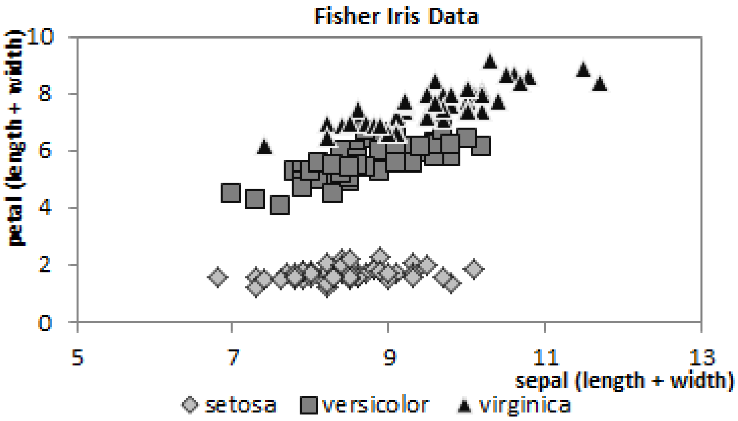


Figure B.3: Reduced Fisher Iris Data

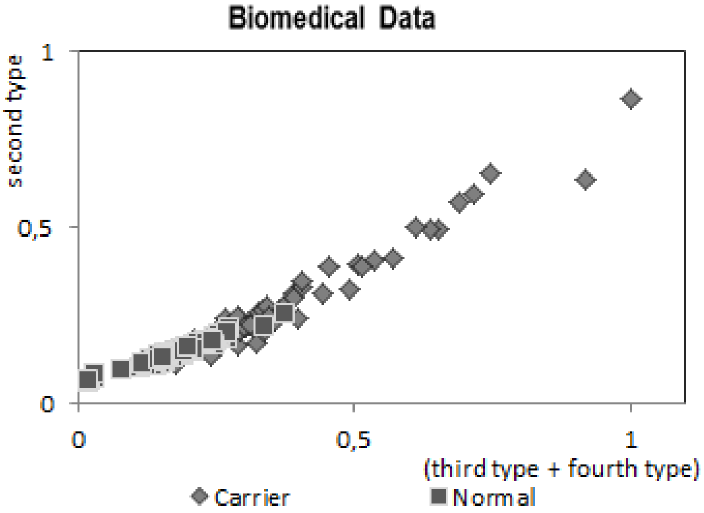


Figure B.4: Reduced Biomedical Data

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Zafer Ataser

Nationality: Turkish (Turkish Republic of North Cyprus)

Date and Place of Birth: 19 Sep 1974, Kıbrıs

Marital Status: Married

email: zafer.ataser@ceng.metu.edu.tr

EDUCATION

Degree	Institution	Year of Graduation
MS	Hacettepe Uni. Computer Engineering	2000
BS	Hacettepe Uni. Computer Science & Eng.	1998
High School	Lapta Yavuzlar High School	1993

WORK EXPERIENCE

Year	Place	Enrollment
1998-2001	Hacettepe Uni. Computer Science & Eng.	Research Assistant
2001-2004	Emek Bilişim	Software Engineer
2004-2006	Girne American Uni.	Senior Lecturer
2006-Present	Kuzey Kıbrıs Turkcell	Database and Management Systems Specialist

FOREIGN LANGUAGES

English

PUBLICATIONS

1. Z. Ataser and F.N. Alpaslan. "Self-Adaptive Negative Selection Using Local Outlier Factor". Computer and Information Sciences III (27th International Symposium on Computer and Information Sciences), 161-169, 2013.