INTRUSION DETECTION SYSTEM USING ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINE: A COMPARATIVE STUDY

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

AISHWARYA MISHRA
MZU REGD. NO. 153 of 2015
PH.D REGD. No. MZU/Ph.D./741 of 11.05.2015

GREATER DEEDS REMAIN
ESTD MIZORAM UNIVERSITY 2001

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
SCHOOL OF PHYSICAL SCIENCES

MARCH
2021

INTRUSION DETECTION SYSTEM USING ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINE: A COMPARATIVE STUDY

BY

Aishwarya Mishra

Department of Mathematics and Computer Science

Prof. Jamal Hussain
Supervisor

Submitted

In partial fulfillment of the requirement for the Degree of Doctor of Philosophy in

Computer Science of Mizoram University, Aizawl.

CERTIFICATE

This is to certify that the thesis entitled, ‘**Intrusion Detection System Using Artificial Neural Network and Support Vector Machine: A Comparative Study**’ submitted by **Aishwarya Mishra**, having Registration No. MZU/Ph.D./741 of 11.05.2015 to the Mizoram University for the Degree of Doctor of Philosophy in Computer Science has been completed under my guidance and supervision. The work done by the candidate is the original one and it has not been submitted to any other university or institution for the award of any degree or diploma and it is within the area of registration.

(**Prof. Jamal Hussain**)
Supervisor
Department of Mathematics and
Computer Science
Mizoram University

Mizoram University

March 2021

DECLARATION

I, Aishwarya Mishra, hereby declare that the subject matter of this thesis is the record of work done by me, that the contents of this thesis did not form the basis of the award of any previous Degree to me or to do the best of my knowledge to anybody else, and that the thesis has not been submitted by me for any research degree in any other University/Institute.

This is being submitted to Mizoram University for the degree of Doctor of Philosophy in Computer Science.

(Aishwarya Mishra)

Head Supervisor

Acknowledgments

At the outset, I extend my prayer before **The Almighty** for blessings on me the strength and energy for the successful completion of the work. I would like to extend my thanks to my friends for boosting me with moral support and help at every stage in the successful completion of the work.

It is a great pleasure to express and acknowledge my esteemed, sincere, noteworthy, and deep sense of gratitude to my guide **Prof. Jamal Hussain**, Professor, Department of Mathematics and Computer Science, School of Physical Sciences, Mizoram University, Aizawl for his constant supervision and stepwise guidance during the entire work. I am indebted to such a dynamic teacher as, without his persistent effort, constructive views, patient, hard work, careful corrections in each chapter, the present work would not have been completed in time.

I am equally grateful to **Dr. Jay Prakash Singh,** Head, Department of Mathematics and Computer Science, School of Physical Sciences, Mizoram University, Aizawl for his support and encouragement.

I am extremely grateful to **Prof. R. C. Tiwari**, Ex-Dean of the School for his blessings and motivation, encouragement to complete the work. I am also grateful to the other faculties of the department for their constant motivation to bring success to the work.

I also equally extend my heartfelt thanks and indebted to **Prof. R. P. Tiwari,** Ex-Finance Officer, Mizoram University for his motivation, noteworthy and consistent encouragement to complete the research work.

I also express my sincere gratitude and obligation to **Ms. Neelam Rout** for her sincere help in bringing success to the work. Her motivation and guidance made me complete the work on time.

I express my deep sense of obligation and gratefulness to my husband **Mr. Mrutyunjay Mishra**, daughter **Ms. Krishanya Mishra** and father-in-law **Mr. Prasanna Kumar Mishra** and others in my family for their constant guidance, encouragement, motivation, support, and untiring help to complete the work.

Last, but not the least, I would like to acknowledge my father **Prof. R. N. Mishra**, my mother **Mrs. Suchitra Mahapatra** for their never-ending blessings, constant support, and a source of inspiration to complete my thesis. I owe my family for the strength, confidence, encouragement, and love rendered to me. My exclusive appreciation goes to my younger sister **Ms. Samikshya Mishra** for her worthy contributions to the successful completion of the work.

Date: (Aishwarya Mishra)
Place : Mizoram University

Table of Contents

List of Figures

List of Tables

List of Graphs

List of Abbreviations

Abbreviation	
A -Module	Analysis Module
ACM	Association of Computing Machinery
ANN	Artificial Neural Network
AUC	Area under Curve
CDS	Credential Service Provider
CFBP	Cascaded Forward Back Propagation
CIA Triad	Confidentiality, Integrity and Availability Triad
D- Module	Database Module
DARPA	Defense Advanced Research Projects Agency
DDOS	Distributed Denial of Service
DECR-RBFN	Decremental Radial Basis Function Neural Network
DIDS	Distributed Intrusion Detection System
DLL	Dynamic Link Library
DNA	Deoxyribonucleic acid
DNS	Domain Name System
DNS	Domain Name Service
DOS	Denial of Service
E -Module	Event Module
EAA	Enterprise Applications Administration
EBP	Elman Back Propagation
EVRBFN	Evolutionary Radial Basis Function Neural Networks
FC-ANN	Fuzzy Clustering Artificial Neural Network
FFNN	Feed Forward Neural Network
FMS	File Monitoring Systems
FSM	File System Monitoring
G-Mean	Geometric Mean
GRNN	Generalized Regression Neural Network
HIDS	Host-based Intrusion Detection System
HTTP Flood	HyperText Transfer Protocol Flood

ICANN	Internet Corporation for Assigned Names and Numbers
ICMP	Internet Control Message Protocol
IDES	Intrusion Detection Expert System
IDPs	Intrusion Detection and Prevention Systems
IDS	Intrusion Detection System
IDWG	Intrusion Detection Working Group
IOCTA	Internet Organized Crime Threat Assessment
IP	Internet Protocol
IPAS	Internet Protocol Address Space
IPS	Intrusion Prevention System
ITU	International Telecommunication Union
KDD	Knowledge Discovery in Data Mining
KEEL	Knowledge Extraction based on Evolutionary Learning
K-FCV	K- Fold Cross-Validation
KKT	Karush-Kuhn-Tucker
LAN	Local Area Network
LDA	Linear Discriminant Analysis
Logsig	Log-Sigmoid transfer function.
LVQ	Learning Vector Quantization
MATLAB	Matrix Laboratory
MCC	Matthews’s Correlation Coefficient
MLFF	Multi-Layer Feed-Forward Neural Networks
MLP	Multilayer Perceptron
MLP-BP	Multilayer Perceptron with Back-propagation
NBA	Network Behaviour Analysis
NIC	Network Interface Card
NIDS	Network-based Intrusion Detection System
NOF	Neighborhood Outlier Factor
NSL-KDD	Network Socket Layer- Knowledge Discovery in Data Mining
NTA	Network Traffic Analysis
NTP	Network Time Protocol

OA-LS-SVM	Optimum Allocation-based Least Square Support Vector Machine
OSI	Open System Interconnection
PCA	Principal Component Analysis
PNN	Probabilistic Neural Network
PPV	Positive Prediction Value
Purelin	Pure linear neural Transfer Function.
R -Module	Response Module
R2L	Remote to Local
RAT	Remote Access Trojan
RBFN	Radial Basis Function Network
RBFN	Radial Basis Function Neural Network
RLSVM	Robust Lagrangian Support Vector Machine
RFC	Request for Comments
ROC Curve	Receiver Operating Characteristics Curve
ROLALSVM	Robust Online Learning Algorithm with Lagrangian Support Vector Machine
Satlin	Saturating linear transfer function
SIEM	Security Information and Event Management
SMW	Sherman-Morrison Woodbury
SOFM	Self-Organizing Feature Map
SOM	Self-Organizing Map
SONN	Self-Optimizing Neural Networks
SRM	Structural Risk Minimization
SSO	Site Security Officer
SVM	Support Vector Machine
SYN Flood	Synchronization Flood
Tansig	Tan-Sigmoid transfer function
TCP	Transmission Control Protocol
TNR	True Negative Rate
TPR	True Positive Rate
U2R	User to Root

UDP	User Datagram Protocol
URL	Uniform Resource Locator
VAIDS	Vulnerability Assessment based Intrusion Detection System
WEKA	Waikato Environment for Knowledge Analysis

CHAPTER-1

INTRODUCTION

Chapter-I

Introduction

1.1 Introduction

The spectacular growth of computer technology and its application through networking in multidimensional sectors such as communication, banking, insurance, industry, education, agriculture, medical, engineering, etc. triggered society in bringing sustainable growth and development in finance, business, knowledge, etc. Further, the massive increase of computers and computer networks use and an immense addition in the number of multifarious coverings on Internet platforms coupled with the revolution in communication technology drastically revamped the information domain. The networking of computers which practically facilitate sharing of data, information, exchange of ideas, etc. operates through the Internet and it is considered to be a global system where, innumerable networks irrespective of the types such as, private, public, academic, government, business sectors are linked together and facilitate to access enormous array information sources for various academic pursuits and other transactions and in the process revolutionised the communication system. Internet, being the most panoptic and open domain allows everybody to browse and access millions of multifarious information causing thereby threat to others. It is due to the non-functional of any centralized governance in either technological implementation or policies for access and usage. Mention may be made that, each constituent network sets its standards. Only the overreaching definitions of two principal namespaces on the Internet, i.e, (i) Internet Protocol Address Space (IPAS) and (ii) Domain Name System (DNS) is directed by maintainer organisations like Internet Corporation for Assigned Names and Numbers (ICANN). This enforces safe and secure transactions of any activities on the Internet platform and it is primarily to abstain from an unforeseen danger, hijack of information on the Internet. Therefore, security has become indispensable to protect the information and other activities carried on the Internet through computers. Security reckons to prevention, protection, damage, invasion of privacy, fraud, unlawful entry in information, literary, physical domain, etc. It does not have any limitations or boundaries. Checking the intruder or the trespassers from attack either physically or technologically or any form has become the call of the hour to safeguard the self and intellectual wealth, knowledge. It has become more pragmatic in the information domain due to the prevalence of technologies in all aspects of life. Given the prevalence of computer-based communication systems in all fields, security has become indispensable which requires measures to protect from unwanted elements. Hence, protection is quite essential to refrain from external intruders in the computer industry. (Stallings and Brown, 2010) viewed it as the techniques for ensuring the protection of data in a computer and traced upon not allow to access without authorization and proper identity. Different types of security measures in a computer such as data encryption and passwords are applied. While data encryption refers to the conversion of data into a form that is opaque without a decoding mechanism the password relates to the secret word or code or phrase. Security can be broadly classified into two types such as,

Active Security: This is principally concerned with a proactive mechanism where security measures are taken in advance to protect the asset which may be literary work or data. Viz, Data Encryption. In other words, it is a phenomenon linked with pre-attack security.

Passive Security: Here, security measures are admitted after the attack on the database, computer system. In other words, this relates to post-attack security.

1.2 Security- The Need

Globalization of information through a computer has become dominant for sharing, use, etc. which precipitated a state of danger, especially in the electronic environment. The danger in the true sense of the term relates to the hijacking of information, breaking down the system, unauthorized access, etc. Hence, security has become crucial which concerns the state of being free from any type of danger or injury of the system encountered illegally. It also connotes taking measures as a precaution against any type of theft, espionage, or sabotage. Bishop and Venkatramanayya (2005) viewed security strategies that can be implanted in three ways such as, (i) Detection, (ii) Prevention, and (iii) Recovery.

1.2.1 Detection

The disturbance created by any means either manually or technically to the system results to break down the normal state of work which needs immediate vigilance to redress the problem. Hence, the need for detection is essential to take preventive measures for the sound functioning of the system. It also relates to any action or process of identifying the presence of disturbing elements concealed.

1.2.2 Prevention

Prevention signifies undertaking measures to get redress to the problems that are likely to occur during the functioning of the system. It also denotes the sense of undertaking remedial measures anticipating future problems. Typical mechanisms supervise various aspects of the system including, looking for course of actions or information indicating an attack. The need has been seriously felt in the digital environment due to the massive use of computers for multifarious works.

1.2.3 Recovery

It is an act of regaining or saving something lost. It is a restoration or returns to any former and better condition. It is an action-oriented mechanism to get back the original form which, however, sometimes very difficult due to the failure of the system leading thereby, to loss of information, data, etc. Hence, mechanisms are developed to restore the files, data, and information through relevant software, hardware, etc.

1.3 Computer Security- The Concept

Computer security connotes to the protection yielded to an automated information system for attaining the desired targets of upholding the integrity, obtainability, and secrecy of information system resources comprising software, hardware, firmware, data, and or information including telecommunications (Stallings and Brown, 2010). Computer security has become obligatory because of the prevailing of various activities such as data interchange, e-mail, web services, and various online operations on the Internet domain. This has become more pragmatic in an automated information system to defend the information through confidentiality, integrity, and availability, etc. which are recognized as the key objectives of the heart of computer security.

1.3.1 Confidentiality

It is related to the secrecy of information or resources. Maintaining confidentiality of information has been seriously felt necessary while using computers in sensitive fields like, defense, government records, database, industry, service organizations like a bank, insurance, etc. Access control mechanism supports confidentiality which can be performed through cryptography that scrabbles data to make it incomprehensible (Bishop and Venkatramanayya, 2005). A loss of confidentiality is the unauthorized disclosure of information. Resource hiding is another vital ingredient of confidentiality. Invariably, many websites conceal the configuration of their configurations and the system as well. Likewise, many organizations also do not reveal the equipment details reason being to avoid unauthorized access. Confidentiality has two related concepts as Data Confidentiality and Privacy.

1.3.1.1 Data Confidentiality

Data and or information exclusively personal and confidential are neither shared nor disclosed to others. The access control mechanism also conceals the existence of data for its protection.

1.3.1.2 Privacy

It is concerned with individual personal information who determines the right persons to unwrapping the information or furnishes the data.

1.3.2 Integrity

Fidelity of data or resources resembles integrity which otherwise means to prevent improper or unlawful change. Integrity can be categorized into two types such as (i) Data Integrity and (ii) System Integrity where the former relates to the content of the information and the latter is concerned with the source and these are crucial to protect the data in the computer and prevent unwanted and unauthorized access. Further, integrity mechanism can be grouped into two different classes such as (i) Prevention Mechanism, and (ii) Detection Mechanism (Bishop and Venkatramanayya, 2005) where the former is associated with the change of data and or information in an unauthorized manner while the latter corresponds to alter the data differently even if he is authorized to perform. Moreover, the detection mechanism does not forbid violations of integrity but alerts that the data integrity is no longer trustable. It, however, analyze the system events to find out problems and reports that the file is corrupt.

1.3.2.1 Data Integrity

Data integrity connotes conserving and assuring the consistency and uniformity of data over its orbit which is significant for design, development, implementation, its usage, storing, processing and retrieval in the interesting field of research, learning. This also ensures maintaining quality in the database. Here, an alteration in data, programs, if any is performed by the authorized person. Data integrity is covered under the categories of (i) Domain Integrity, (ii) Entity Integrity, (iii) User-defined Integrity, and (iv) Referential Integrity.

1.3.2.2 System Integrity

It assures the performance of a system with the intended function which is free from any premeditated or inadvertent unauthorized manipulation.

1.3.3 Availability

It ascertains the availability of information instantly and does not deny providing service to the authorized individual over the system.

Inclusively, the term Confidentiality, Integrity, and Availability are identified as CIA Triad which is explained in the following Figure-1.1

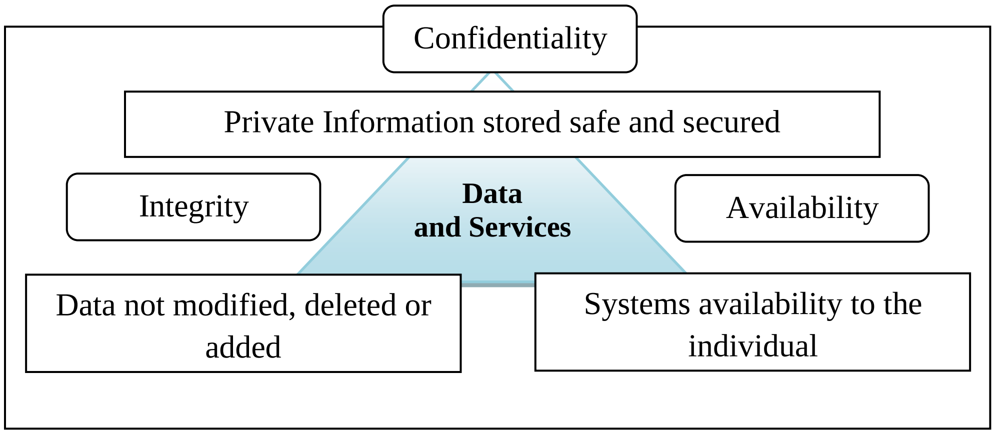

Confidentiality
Private Information stored safe and secured
Integrity
Data and Services
Availability
Data not modified, deleted or added
Systems availability to the individual

Fig 1.1: CIA Triad

However, in addition to the above three securities of the computer, another two types of securities are also equally important such as, Authenticity and Accountability, which are explained below.

1.3.4 Authenticity

The property which is real or valid and not fake is known as authenticity. Further, it is tested and relied on with confidence in the validity of a transmission, a message, or a message originator. This otherwise means that the user confirms the receiver that each input arriving at the system hails from a reliable source.

1.3.5 Accountability

This connotes the security goal that generates the requirement for actions of an entity that requires tracing uniquely to that entity. This supports intrusion detection and prevention, non-repudiation, deterrence, fault isolation, including after-action recovery and legal action (Kissel, 2013).

Data security is a global concern and hence, international agency, especially in the industry sector, has developed advanced threat protection portfolios of products and solutions. It could be visualized from the initiatives taken by Cisco who developed a threat-centric and operational approach to security intending to trim down the complexity with relentless vigilance and control over the network environment.

Having said that, the proliferated use of the Internet resulted in multiple functions such as e-commerce, online transactions, e-mail, information retrieval, Electronic Data Interchange, File Transfer, and Web-services, etc. and in a parallel way of development, threats also equally prevailed leading to defunct/corrupt of the system. This, however, has become crucial to prevent threats as it is a potential violation of security. Further, the unscrupulous elements with iniquity motives play a dynamic role to obliterate and defunct the system in the network environment. Such attempts are recognized as attacks and those who involve in such unscrupulous activities are known as attackers. Hence, it is pertinent to ascertain security attacks. Over and above, other terms are also related to computer security. The relationships among other terms associated with computer security are discussed in Figure-1.2.

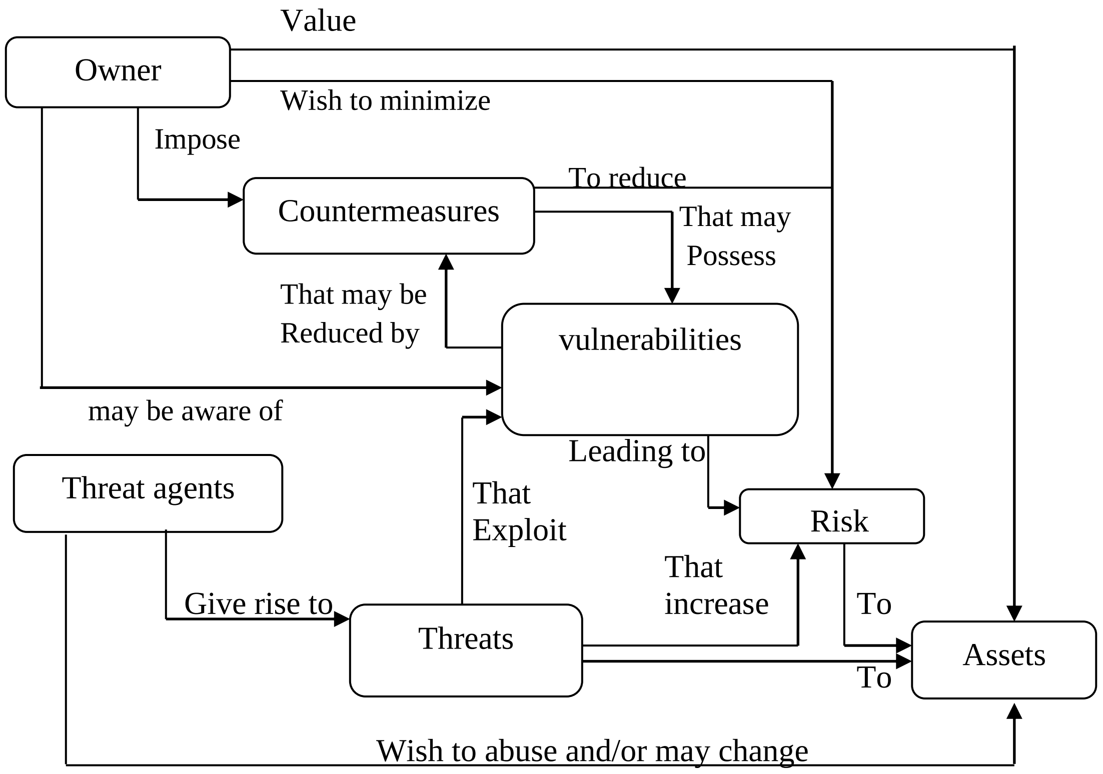


Fig. 1.2: Security Concepts and Relationship
Source: Stallings and Brown, 2013.

1.4 Security Attacks

Security attack resembles an action that compromises the security of information possessed by an organization, institution, individual, company, etc. and it is one of the components of Open System Interconnection (OSI), a security architecture developed by the International Telecommunication Union (ITU), a United Nations origin in the context of the OSI protocol architecture (Stallings, 2011). The security attacks according to Stallings (2011) are classified into two types such as, (i) Passive Attacks and (ii) Active Attacks which are used both in X.800 and RFC 2828. The different types of security attacks (Anwar *et al.*, 2017) are discussed in *Appendix-I.*

1.4.1 Passive Attacks

A passive attack is associated with an effort to use the system for information retrieval. Further, it is an attack against a certification protocol where the assailant bugs data traveling along with the network between the Claimant and Verifier without data alteration which is otherwise known as eavesdropping (Kissel, 2013). The passive attack is grouped into two types such as (i) Release of Message Contents and (ii) Traffic Analysis.

1.4.1.1 Release of Message Contents

The telephone conversation, message through both mail and mobile, transferred file, etc. contain the messages which are delivered to the recipient on the Internet or other communication system and the attacker come across with the information.

1.4.1.2 Traffic Analysis

Traffic analysis is one type of passive attack where an intruder keeps a track of information about calls including messages and locates the origin and destination numbers, or frequency and length of the messages. The common technique applied for masking the contents is encryption but, despite encryption, the attacker still gets through the message patterns and determines the location and identity of the sender including the frequency and length of the message. The attacker here can conjecture the message as the message is encrypted. In a network environment, it finds difficult to detect the attacker who resonates en-route in the communication channel as the attacker does not alter the message and or data. Hence, emphasis requires lying down on prevention through encryption than detection in network surroundings.

1.4.2 Active Attacks

Active attack resembles modification, alteration of the data and or information en-route while sending the message or information by the sender to the receiver. It amounts to a change of the data stream or generation of a false stream. It is an attack on the authentication protocol where the attacker transmits data to the claimant, Credential Service Provider (CDS), verifier, or relying party. Active attacks can be categorized into four types such as (i) Masquerade, (ii) Replay, (iii) Modification of Messages, and (iv) Denial of Service (Stallings, 2011).

1.4.2.1 Masquerade

It signifies to gain access to a system or performs a malicious act by an unauthorized entity by posting as an authorized entity. Masquerading, which is also known as spoofing is an impersonation and it is a clear violation of security. Invariably it is materialized when an attacker accesses the system with a confirmed user login ID and password of another user. Another example of masquerade is malicious logic such as, Trojan horse which appears to perform a useful or desirable function but gains illegal access to a system.

1.4.2.2 Replay

An attack involves capturing transmitted authentication or access control information of others' systems and retransmission to produce an unauthorized effect or gaining unauthorized access. Here, a user uses a transmission channel i.e., the Internet while sending a message or information to the receiver which is captured by an attacker en-route and later, the attacker replays the message or information and sends it to the original receiver.

1.4.2.3 Modification of Messages

It pertains to an alteration of some portions of the message by the attacker en-route while sending the message by the sender to the receiver using a communication channel which may give rise to delay in sending the message with an unauthorized effect to the original message. In other words, when the sender sends the message to the target receiver using either Internet or other communication channel is abducted by the attacker who later on fabricates the message and transmits it to the destination and it affects the erroneous functioning.

1.4.2.4 Denial of Service (DoS)

Denial of Service is the prevention of authorized access to resources or the delaying of time-critical operations. Here, depending on the service provided the time-critical may be milliseconds or hours. This attack may include a specific target where an entity may suppress all messages directed to a particular destination. Another form of service denial relates to the dislocation of the entire network which can be performed either by crippling the network or by clogging it with multiple messages leading to degrading performance (Stallings, 2011).

Hence, from the above discussions, it could be found that DoS is an attack in which a user or an organization has deprived the resources on the network by flooding the network with useless traffic (NIIT, 2004). Further, DoS may be launched either from a single computer or a group computer distributed on the Internet which is known as Distributed Denial of Service Attack (DDoS).

1.4.2.4.1 Denial of Service (DoS) Attack

A Denial of Service (DoS) attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service.

Synchronization Flooding known as SYN flooding is a typical and effective technique used by DoS attacks. Like SYN flooding, the smurf attack is also another typical type of DoS attack where smurf is used to execute the attack. It sends unreasonable messages to the target computer and crashes by consuming all its resources. The attacker in a typical smurf sends crafted ping requests to many computers within a minimum time where the source IP address in the crafted ping request is placed with the victim's IP address. Figure-1.3 placed below clearly depicts the Smurf Attack.

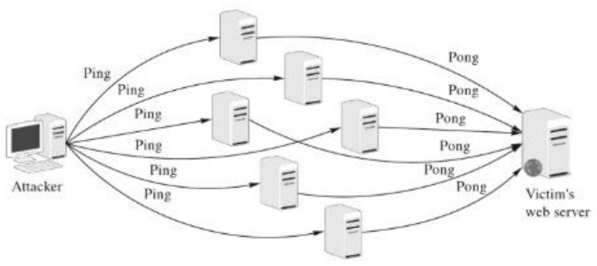


Fig. 1.3: Smurf Attack
Source: Wang and Kissel (2015)

Hence from the foregoing discussions, it could be ascertained that Denial of Service (DoS) attacks invariably interrupt the network service and the system. The DoS attack broadly can be categorized into three types such as (i) Host-based Attack, (ii) Network-based Attack, and (iii) Distributed based Attack has been discussed below.

- **Host-based Attack**

It is primarily associated with attacking the operating system, application software, or the configuration of the targeted host system. Grabbing resources on a host system is the possible way of DoS.

Resource hogging that comprises CPU time and memory use is a possible way and most common target of DoS on a host system. The crashers, a form of host-based DoS is designed to collapse the host system. The crashers with an evil intention apply their motives not only to target a vulnerability in the host system but also exploit the implementation of network protocols through operating systems. The operating systems in such an environment may not accept certain packets which, however, on receipt of the same hang or crash the operating system.

- **Network-based Attack**

  Denial-of-service attacks are executed very frequently against network connectivity to forbid hosts or networks from communicating on the network. SYN flood attack is one type of network-based attack. The attacker in such type of attack takes initiatives of establishing a connection to the victim system computer cunningly to prevent the ultimate completion of the connection. The attacker through network-based DoS attacks targets the network resources and upsets the legitimate use. The network-based DoS flood the network and the target system with immense packets which seriously affect the bandwidth of the legitimate user. Three important and dangerous methods of flooding have identified that include,

  - TCP Floods- Here, Transmission Control Protocol (TCP) packets teem to the target system;
  - ICMP Echo Request/Reply 6- ICMP (Internet Control Message Protocol) packets are flow profusely to the target system;
  - UDP Floods- User Datagram Protocol (UDP) packets are welled out to the target.

- **Distributed based Attack**

  The distributed-based attack is very much associated with the Distributed Denial of Service (DDoS) attack. Mention may be made that, the DDoS attacks equally add a massive problem on the Internet which is a comparatively simple but very powerful technique to attack the internet resources. It attacks substantially attach many-to-one-dimension to the DoS problem which becomes much problematic. There are no apparent characteristics of DDoS streams that could be directly used for their detection. This DDoS attack influences the User Datagram Protocol (UDP) which is a session layer networking protocol. The DDoS attack floods random ports to a remote system with enormous UDP packets leading thereby, causing the remote system to repeatedly check for the application listening at that port, and in the event of detecting no application, it replies with an ICMP Destination Unreachable packet. This procedure fluids host resources which lead to inaccessibility.

Further, DDoS attacks that constitute packets streams from heterogeneous sources engage a good number of coordinated systems on the Internet for consuming some decisive resources at the target and restrain the service to genuine clients. In the process, the traffic is aggregated in such a way that there lay constraints to differentiate the genuine packets from attack packets which are voluminous. This leads the system difficult to control the packets. Hence, in absence of precautionary measures which needs to be applied, DDoS victim encounters with substantial impairments ranging from system shutdown and file corruption to total or partial loss of services (Douligeris and Mitrokotsa, 2004).

- **DDoS Attack**

The Distributed Denial of Service attack moves according to the following sequence.

i. It compromises a maximum number of networked computers which may be achieved using Trojan horses.

ii. It installs special software on the compromised computers to accomplish a DoS attack later on. The installed software is known as *zombie software* and the system on which it is installed is called a *zombie computer* or a *zombie.* A group of *zombies* is called a *zombie army* which is typically identified as a *botnet.*

iii. An attack command is issued to every *zombie* computer for launching a DoS attack on the same target at the same time.

1.4.2.4.2 Distributed Denial of Service (DDoS)

While tracing a brief genealogy, Distributed DoS (DDoS) attacks in computer and network domain were first detected in 1999 while introducing attack tools such as ‘The DoS Projects Trinoo’ (Axelsson, 2000; Cohen, 1995), The Tribe Flood Network

(Axelsson, 2000; McLuhan, 1987) and Stacheldraht8 (Cohen, 1995). DDoS attacks materialize into action by using a large number of attack hosts and gives directions for a simultaneous attack on a target or targets. To start the process of DDoS attack, the attacker exercises command to the master system to start the attack which is extended to all demon nodes under them to affect the attack, and thereafter, the affected nodes aggress the target system causing thereby, a denial of service to the victims' system. With enough daemon nodes, even a simple web page request will forbid the target from serving legitimate user requests. The DDoS assault takes place when several compromised systems are infected employing the malicious code act simultaneously and are coordinated beneath the manipulate of a single attacker to crash into the victim's system and wear out its resources.

There are some other specific, popular, and dangerous types of DDoS attacks which are discussed below in Table-1.1 include ICMP (Ping) (Deshmukha and Devadkarb, 2015).

Table-1.1- Types of DDoS attacks

ICMP (Ping) Flood	Internet Control Message Protocol (ICMP) which is an error-reporting protocol network device uses creating error messages to the source IP address when network problems forbid the delivery of IP packets. User Datagram Protocol (UDP) flood attack overpowers the target resource with ICMP Echo Request (ping) packets and sends packets at a quick rate without waiting for replies.
SYN Flood	A Synchronise (SYN) flood DDoS attack exploits a known weakness in the TCP connection sequence (the “three-way handshake”), wherein an SYN request to start a TCP connection with a host must be answered by a Synchronise-Acknowledge (SYN-ACK) response from that host, and then confirmed by an ACK response from the requester.

Ping of Death	The attacker explores measures through a Ping of Death ("POD") attack by sending numerous malformed or malicious pings to a system. In a Ping of Death, the recipient ends up with an IP packet with malicious manipulation of fragment contents.
Slowloris	Slowloris is a highly-targeted attack that enables one web server to take down another server without disturbing other services or ports on the target network. Slowloris repetitively sends more HTTP headers but never ends a request.
NTP Amplification	The executor of Network Time Protocol (NTP) Amplification attacks exploits publically-accessible NTP servers to overwhelm the targeted server with UDP traffic
HTTP Flood	The attacker in Hyper-Text Transfer Protocol (HTTP) flood DDoS attack exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. The attack is more intensified during the allocation of maximum resources in the server or application in response to every single request.

Source: Deshmukh and Devadkarb, 2015

Apart from the above types of DoS and DDoS attacks, other types of attack as discussed below also prevail in the network environment which defunct the systems and it includes, (i) Distributed Attack, (ii) Insider Attack, (iii) Close-in- Attack, (iv) Phishing Attack, (v) Hijack Attack, (vi) Spoof Attack, (vii) Buffer Overflow, (viii) Exploit Attack, (ix) Password Attack that comprises Dictionary Attack, Brute-force Attack, and Hybrid Attack. The detailed comparisons of various types' attacks with examples (Anwar et al., 2017) have been illustrated in *Appendix-II.*

1.5 Principles of Security

There are five stages of security principles as discussed below to compromise intrusion detection. These five steps appear to be the phases of activity through which the attackers make transitions (Bejtlich, 2005).

1.5.1 Reconnaissance

Reconnaissance which is technical and as well as non-technical is associated with the process of validating connectivity, enumerating services including checking of vulnerable applications. The intruder verifies the vulnerability of service before exploitation and successfully exploits the target. Structured threats typically select a specific victim and then perform reconnaissance to devise means of compromising their target. Reconnaissance helps structured threats to plan their attacks in an efficient way and unnoticeable manner.

1.5.2 Exploitation

In computing, an exploit is an attack on a computer that takes advantage of a vulnerability that the system offers to intruders. The term refers to the act of a successful attack on a system. It is a process involved in abusing or breaching services on a target. Further, abuse of service is a concern with the involvement of making illegitimate use of a legitimate mode of access.

1.5.3 Reinforcement

Reinforcement is the stage where the intruder takes advantage of unauthorized access to gain additional capabilities on the target. While some exploits yield immediate remote root-level privileges, some provide only user-level access. The attackers find a way to elevate their privileges and put those illegal gains to work. At this point, the intruders have the root control over both their workstations and those of their victims.

1.5.4 Consolidation

In computing, consolidation denotes data storage or server resources that are apportioned among many users and accessed by numerous applications. Its ambitions are to make environment-friendly use of computer assets and forestall servers and storage tools from being under-utilized and taking too an awful lot of space. Consolidation occurs when the intruder communicates with the victim server through the back door which takes the form of a listening service to which the intruder connects. It could be a stateless system relying on the sequences of specific fields in the IP, TCP, UDP, or other protocol headers.

1.6 Threat

Threats in computer especially in a network environment have become accustomed phenomena which prevent the system from a common behaviour. The term has been defined by National Information Assurance Glossary as ‘Any circumstance or event with the potential to adversely impact an information system through unauthorized access, destruction, disclosure, modification of data, and/or denial of service’. The National Institute of Standards and Technology Interagency elucidated that, a threat refers to as any circumstance or match with the potential to adversely have an impact on organizational operations inclusive of mission, functions, image, or reputation, organizational assets or persons through a record of the gadget by unauthorized access, destruction, disclosure, modification of information, and/or denial of service. The achievable for a threat-source also successfully exploit specific information system vulnerability (Kissel, 2013). Hence, from the foregoing discussions, it could be observed that the threat is considered as a violation of the normal way of functioning of the system. It has become more prevalent on the internet domain for a computer system while on-line operations are carried out. The threat on the Internet creates a platform for unauthorized monitoring and intrusion in the network traffic causing vulnerabilities and incidents to the system.

The internet vulnerabilities comprise security weaknesses in both Windows and Linux-based operating systems of the working computers. This also includes Internet routers and other network devices. These types of vulnerabilities comprise denial-of-service attacks, IP spoofing, passwords sniffing, etc. where the intruders generate packets with false IP addresses and exploit its applications that use authentications based on IP and various forms listening in and packet sniffing (Stallings and Brown, 2010).

1.6.1 Divisions of Threats

Broadly, the threats can be classified into two groups such as i) Unauthorised Users Accessing Role Accounts and ii) Authorised Users Accessing Role Accounts.

- **Unauthorised Users Accessing Role Accounts**

The attackers behave negatively and are indulged in nefarious activities that cause harm to others intentionally. They access the other computers in a network platform and acquire the personal data and exploit such a condition that crashes the system and or inserts crafted code to gain control over the system of others. This type of unauthorized access can be classified into four classes.

❶ An unauthorized user identified as a trespasser may access to a role account;

❷ An authorized user may use an insecure channel to access a role account, thereby revealing authenticate information to an unauthorized person in a remote location;

❸ An unauthorized user may modify the access control to gain access to the role account;

❹ An authorized user may execute a Trojan horse or other form of malicious logic giving an unauthorized user access to the role account. It may be mentioned that malicious logic constitutes the virus programs like, (a) Rabbits and Bacteria which absorb all or some resources and (b) Logic Bombs that operate a motion violates the safety coverage when some external incident occurs

- **Authorized Users Accessing Role Accounts**

  The threats relate to an authorized user changing access permissions or executing unauthorized commands can be explained as follows.

- An authorized user may obtain access to a role account and perform unauthorized commands;
- An authorized user may execute a command which performs functions where the user is unauthorized to perform;
- An authorized user may change the restrictions on the users' ability to obtain access to the account (Bishop and Venkatramanayya, 2005).

1.6.2 Security Threats

The security threats can be categorized as under which corresponds to the meaning as explained below against each type. The various security threats have been described below.

- ➔ Interruption- It is an attack on availability;
- ➔ Interception- It concerns an attack on confidentiality;
- ➔ Modification- It relates to an attack on integrity; and
- ➔ Fabrication- It is associated with an attack on authenticity.

1.7 Intrusion Detection

A security model that monitors the network traffic for suspicious activity on the Internet is referred to as intrusion detection and it operates as a model (Denning, 1987 and Kemmerer *et al.*, 2002). It issues alerts the users while such malicious activities are detected in the system. It examines events in network traffic; operating systems etc. and arouses alarm when the events are suspected to be the intrusion symptoms (Sommestad and Hunstad, 2013). A set of the historical profile of the users which is maintained by the intrusion detection model matches the appropriate profile with audit records and update the same in the event of any change in the profile followed by reports in case of detection of an anomaly (Wua and Yen, 2009 & Cort, 2004).

1.7.1 Intrusion Detection System

An intrusion detection system is a defense mechanism that identifies the malicious action targeted at computing and networking resources. While tracing the history, James A Anderson in 1980 conceived the idea of computer security threat and developed monitoring and surveillance to detect malicious activity using event tracking records or audit logs followed by Dorothy Denning and Peter Neumann (1985) developed a prototype model for a real-time Intrusion-Detection Expert System (IDES). Amorso (1999) viewed the intrusion detection system as a process of identifying and responding to malicious activity in computing and networking resources. The security mechanisms of a computer system must be designed so that it is much capable to stop unauthorized admittance to system resources and data. However, at present getting prevented the breaches of security appear is impractical. But efforts are initiated to detect these intrusion attempts to repair the damaged system. (Kashyap *et al.,* 2013 & Alsadhan and Khan, 2013).

There are three different evaluation criteria in the Intrusion Detection System such as. (Debar *et al.,* 1999)

1. Accuracy	–	Relates to the occurrence of false-positive incidents.
2. Performance	–	Processing rate reckons on employed algorithms and tools.
3. Completeness	–	Connotes to the occurrence of false-negative incidents.

1.8 Existing Techniques of Intrusion Detection System

Invariably, two types of techniques such as (i) Anomaly detection and (ii) Misuse detection also known as signature detection are employed in intrusion detection systems (Cannady, 1998). While anomaly detection explains the abnormal behaviour pattern, misuse detection focuses on the use of known patterns of unauthorized behaviour. Growing complexities and the number of attacks, the machine learning technique is left out as the only option for building and maintaining an anomaly detection system with the least human intervention (Bruha, 2000). It may be mentioned that the machine learning technique in intrusion detection built automatically the model based on a training dataset which contains a collection of data instances and using a set of attributes, each of the data is described as categorical and continuous (Kayacik *et al.*,2005). Further, the labels associated with data instances are in binary form i.e, normal and anomaly where, the anomaly is assorted with different attack types such as DoS, U2R, R2L, and Probe. Supervised Anomaly detection provides a better detection rate as compared to the unsupervised method (Vinueza *et al.*, 2004). The supervised learning method comprises Fuzzy logic, neural network, support vector machine, decision tree, Bayesian network, etc. (Acid *et al.*, 2003).

1.9 Artificial Neural Network (ANN)

Artificial Neural Network is an adaptive parallel distributed information processing model that consists of the simple processing unit or neurons, a set of synapses, the network architecture, and a learning process to train the network. It also consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. Artificial Neural Network is also an adaptive system that can capable of changing its structure based on external or internal information that flows through the network during the learning process. (Ryan *et al.*,1997). The processing elements are the fundamental building block of Artificial Neural Network. These elements are responsible for all the computations that are taking place inside the network. The Artificial Neural Network can be broadly categorized into (i) Feed Forward Network and (ii) Recurrent Network. The learning process of Artificial Neural Network is broadly divided into two categories such as (i) Supervised (labeled) and (ii) Unsupervised (unlabelled) (Gorbani *et al.*, 2010).

1.9.1 Artificial Neural Network in Anomaly Detection

Artificial Neural Network, one of the oldest systems is a globally admired mechanism for Anomaly Detection. It comprises various techniques to detect intrusion. Generally, the techniques are classified into three categories such as, (i) Supervised

ANN-based intrusion detection, (ii) Unsupervised ANN-based intrusion detection, and (iii) Hybrid ANN-based intrusion detection having different applications domain. The application of Supervised Artificial Neural Network to Intrusion Detection System is concerned with Multi-Layer Feed-Forward (MLFF) Neural Networks, Multilayer Perceptron (MLP), and Recurrent Neural Networks. The application of unsupervised Artificial Neural Network relates to Self-Organising Map (SOM). The hybrid application combines both the supervised and unsupervised Artificial Neural Network to detect intrusion (Sodiya *et al.,* 2014). The main drawbacks of Artificial Neural Network into Intrusion Detection System can be explained in two aspects such as (i) Lower detection precision, especially for low-frequent attacks i.e. Remote to Local (R2L), User to Root (U2R), and (ii) Weaker Detection Stability (Kashyap *et al.,* 2013).

1.10 Support Vector Machine

Vapnik (1995) developed the Support Vector Machine (SVM) which is a linear machine. The SVM solves the problems related to classification, learning, and prediction (Kausar *et al.,* 2011). As compared to other classifiers, SVM not only embraces Structural Risk Minimization (SRM) principle, it also evades the local to a minimum, resolves over learning matters, and permits good generalization ability. Further, in a high dimensional space, it performs the classification of the data vectors by a hyperplane or set of hyperplanes. It may be pointed out that, for classification, several hyperplanes can be employed for separation but the best hyperplane brings about maximum margin between the data points of two classes. The data points, in most of the cases, are not separable linearly in the input space. Hence, nonlinear transformations are required into a high dimensional space and thereafter, the linear maximum margin classifier can be applied. Here, Kernel functions are used to accomplish the task. They are employed at the training time of the classifiers to select the support vectors along the surface of the function. Then SVM classifies the data by using these support vectors which outline the hyperplane in the feature space. Support Vector Machine increases the dimensionality of the samples to separate the input data. Support Vector Machine is a maximum-margin hyperplane that places in the same class and it classifies data separated by non-linear boundaries. Support Vector Machine with linear and nonlinear kernels have become one of the most capable supervised learning algorithms and can assemble a nonlinear separating that is implicitly defined by a kernel function (Gorbani *et al.*, 2010).

1.10.1 Support Vector Machine in Anomaly Detection

Support Vector Machine is conveniently used in anomaly detection as it produces accurate classification results even with the small quantum of the dataset. It also provides less overfitting, robust to noise, etc. Support Vector Machine can be applied also for multi-class classification problems i.e, one class against another or one class against all classes.

1.11 KDD Cup'99 Dataset

KDD Cup happens to be the global database of the annual Data Mining and Knowledge Discovery competition organized by the Association of Computing Machinery (ACM) Special Interest Group on Knowledge Discovery and Data Mining which is a leading and significant international organization of data miners (http://www.kdd.org/ kdd-cup). KDD-cup has been associated with various competitions since 1997, a list of which has been mentioned below in Table-1.2 and out of which, KDD-Cup'99 primarily is devoted to Computer Network and Intrusion Detection.

It is a globally accepted data set which is primarily devoted to evaluating anomaly detection and it is prepared by Stolfo *et al.* (2000), and it is developed based on the data captured in Defense Advanced Research Projects Agency'98 (DARPA)'98 Intrusion Detection System evaluation which comprises tcpdump data of 7 weeks of network traffic that can be processed into 5 million connection records each with 100 bytes. Mention may be made that, two weeks of test data constitute 2 million connection records approximately. KDD'99 dataset comprises around 4,900,000 single connection vectors where, each 41 features constitute and labeled as normal or an attack with one specific attack type (Tavallaee *et al.*, 2009 & Aggarwal and Sharma, 2015).

Table 1.2- Annual KDD Cup center with the focused area

Sl.No.	**Competition**	**Focused Area**
1	KDD-Cup 1997	Direct marketing for lift curve optimization
2	KDD-Cup 1998	Direct marketing for profit optimization
3	KDD-Cup 1999	Computer network intrusion detection
4	KDD-Cup 2000	Online retailer website clickstream analysis
5	KDD-Cup 2001	Molecular Bioactivity; plus, Protein locale prediction
6	KDD-Cup 2002	BioMed document; plus, Gene role classification
7	KDD-Cup 2003	Network mining and usage log analysis
8	KDD-Cup 2004	Particle physics; plus, Protein homology prediction
9	KDD-Cup 2005	Internet user search query categorization
10	KDD-Cup 2006	Pulmonary embolisms detection from image data
11	KDD-Cup 2007	Consumer recommendations
12	KDD-Cup 2008	Breast Cancer
13	KDD-Cup 2009	Customer relation prediction
14	KDD-Cup 2010	Student performance evaluation
15	KDD-Cup 2011	Predict music rating and identify favourite songs
16	KDD-Cup 2012 (Track-1)	Predict which users (or information sources) one user might follow in Tencent Weibo
17	KDD-Cup 2012 (Track 2)	Predict the click-through rate of ads given the query and user information
18	KDD-Cup 2013 (Track-1)	Author-Paper identification
19	KDD-Cup 2013 (Track-2)	Identify which authors correspond to the same person
20	KDD-Cup 2014	Predict funding requests that deserve an A+
21	KDD-Cup 2015	Predicting dropouts in Massively-Online Open Course

		(MOOC)
22	KDD-Cup 2016	Whose papers are accepted the most: towards measuring the impact of research institutions

Source: http://shodhganga.inflibnet.ac.in/bitstream/10603/9850/8/08_chapter%203.pdf
http://www.kdd.org/kdd-cup

1.11.1 Classification of Attacks in the KDD'99 dataset

Multiple numbers of attacks are there which penetrate the network domain over a period. Primarily, the attacks can be grouped into four classes as discussed below.

- **Denial of Service (DoS)**

This is concerned with Denial of Service. It is also a type of attack where the hacker builds memory resources too busy to serve the legitimate networking requests and hence, denying users access to a machine. DoS attack initiates in three ways such as by,

(i) Abusing the computer's legitimate features.
(ii) Targeting the implementation bugs.
(iii) Exploiting the misconfiguration of the systems.

Further, the attacker provides different modes of services that are inaccessible by the authentic uses, and based on the same, DoS attacks are classified. Examples of such attacks include apache, smurf, Neptune, ping of death, back, mail bomb, UDP storm, etc.

- **Probe**

It relates to surveillance and other probing in the class. Here, the hacker while scanning a machine or a networking device for determining weaknesses or vulnerabilities that may later be exploited to compromise the system. This technique primarily is associated with data mining viz, satan, saint, portsweep, mscan, nmap, etc.

- **Remote to Local Attack (R2L)**

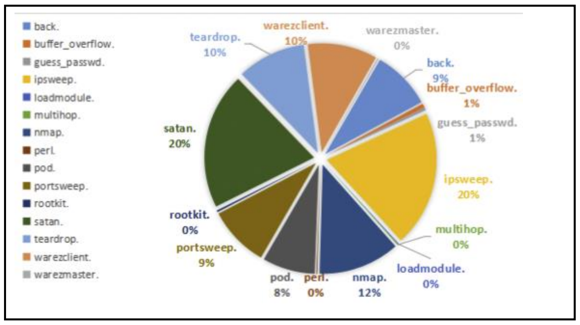
  It pertains to unauthorized access from a remote machine. Here, a user attacks a remotely located machine by sending the packets over the internet and the user does not have access to expose the machine vulnerabilities and exploit privileges that a local user would have on the computer. Examples of such classes fall as xlock, xnsloop, phf, sendmail, dictionary, etc.

- **User to Root Attack (U2R)**

  It is associated with unauthorized access to local superuser (root) privileges. Invariably, these types of attacks are the exploitations where the hacker commences on the system with a normal user account and efforts to abuse vulnerabilities in the system for gaining superuser privileges. Ex- perl, Xtream, etc.

1.11.2 Data Distribution in KDD’99 dataset

The KDD’99 includes a huge number of repeated records of 78% and 75% redundant data on training and test dataset. The redundant dataset can harm the result of the evaluation to a much higher degree of detection accuracy. The data distribution of the KDD’99 dataset is shown in Figure-1.4 below. Tavallaee *et al.* (2009), viewed that, the necessary adjustment made on the KDD’99 dataset results in a dataset known as NSL-KDD. Further, Mchugh (2000) observed that NSL-KDD is also not ideal as it restrains the evaluation result which is due to the use of synthetic simulation of normal with the scripted anomaly. A detailed description of the name, types, mechanisms, and effect of the attack of all 22 types both in normal and other attacks is shown below in Table-1.3. This is coupled with the description of the redundant records of the KDD’99 training and testing dataset in Table-1.4 below (Olusola *et al.*, 2010).



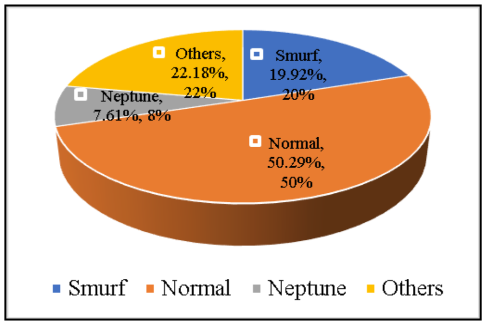


Fig. 1.4 Data Distribution of KDD'99 dataset
Source: https://www.sciencedirect.com/science/article/pii/S2352864817300810#fig6

Table-1.3: Detail Description of various attacks of KDD'99 dataset

Sl. No.	**Name of the attack**	**Type**	**Mechanism**	**Effect of the attack**
1	Back	DoS	Abuse/Bug	Slows down server response
2	land	DoS	Bug	Slows down server response
3	Neptune	DoS	Abuse	Slows down server response
4	Smurf	DoS	Abuse	Slows down server response
5	pod	DoS	Abuse	Slows down server response
6	teardrop	DoS	Bug	Reboots the machine
7	loadmodule	U2R	Poor environment sanitation	Gains root shell
8	buffer_overflow	U2R	Abuse	Gains root shell
9	rootkit	U2R	Abuse	Gains root shell
10	perl	U2R	Poor environment sanitation	Executes commands as root
11	phf	R2L	Bug	Gains user access
12	guess_passwd	R2L	Login misconfiguration	Gains user access

13	warezmaster	R2L	Abuse	Gains user access
14	imap	R2L	Bug	Gains user access
15	multihop	R2L	Abuse	Gains user access
16	ftp_write	R2L	Misconfiguration	Gains user access
17	spy	R2L	Abuse	Gains user access
18	warezclient	R2L	Abuse	Gains user access
19	satan	Probe	Abuse of feature	Looks for known vulnerabilities
20	nmap	Probe	Abuse of feature	Identifies active ports on a machine
21	portsweep	Probe	Abuse of feature	Identifies active ports on a machine
22	ipsweep	Probe	Abuse of feature	Identifies active machines

Table 1.4: Redundant records in KDD'99 training dataset & test dataset

Description	**Training Dataset**		**Total**	**Testing Dataset**		**Total**
	Normal	**Anomaly**		**Normal**	**Anomaly**	
Original Records	972,781	3,925,650	4,898,431	60,591	250,436	311,027
Distinct Records	812,814	262,178	1,074,992	47,911	29,378	77,289
Reduction Rate	16.44%	93.32%	78.05%	20.92%	88.26%	75.15%

Further, the other attacks associated with the KDD’99 dataset as pointed out by Kristopher (1999) have been listed below in Table-1.5 showing their mechanism and consequential effects. The author further mentioned that unauthorized persons in many ways pervert the network or the system to gain access or slow down the response.

Table 1.5: Description of Attacks

Name of the Attack	Type	Mechanism	Effect of the attack
Back	DoS	Abuse/Bug	Slows down server response
land	DoS	Bug	Slows down server response
Neptune	DoS	Abuse	Slows down server response
Smurf	DoS	Abuse	Slows down server response
pod	DoS	Abuse	Slows down server response
teardrop	DoS	Bug	Reboots the machine
loadmodule	U2R	Poor environment sanitation	Gains root shell
buffer_overflow	U2R	Abuse	Gains root shell
rootkit	U2R	Abuse	Gains root shell
perl	U2R	Poor environment sanitation	Executes commands as root
phf	R2L	Bug	Gains user access
guess_passwd	R2L	Login misconfiguration	Gains user access
warezmaster	R2L	Abuse	Gains user access
imap	R2L	Bug	Gains user access
multihop	R2L	Abuse	Gains user access
ftp_write	R2L	Misconfiguration	Gains user access
spy	R2L	Abuse	Gains user access
warezclient	R2L	Abuse	Gains user access
satan	Probe	Abuse of feature	Looks for known vulnerabilities
nmap	Probe	Abuse of feature	Identifies active ports on a machine
portsweep	Probe	Abuse of feature	Identifies active ports on a machine
ipsweep	Probe	Abuse of feature	Identifies active machines

Source: Kendall (1999). A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems.

1.11.3 KDD Cup'99 Features

KDD Cup’99 dataset comprises a total number of 41 features which includes (i) Basic Features comprising of 9 (Nine) individual TCP connections, (ii) 13 (Thirteen) Content features within a connection suggested by domain knowledge, and (iii) 9 (Nine) Traffic features computed using a two-second time window, and (iv) 10 (Ten) Host-based features A comprehensive list of all 41 features are placed below in Table-1.6.

Table 1.6: Features of the KDD'99 dataset

Feature Name	Variable type	Category	Label	Description
Basic Features of Individual TCP connections				
duration	C	1	v1	Number of seconds of the connection
protocol_type	D	1	v2	Type of the protocol, e.g., tcp, udp, icmp etc.
service	D	1	v3	Network service on the destination, e.g., http, telnet, etc.
flag	D	1	v4	Normal or error status of the Connection
src_bytes	C	1	v5	Number of data bytes from source to destination
dst_bytes	C	1	v6	Number of data bytes from destination to source
land	D	1	v7	1-connection is from/to the same host/port; 0-otherwise
wrong_fragment	C	1	v8	Number of ‘wrong’ fragments

urgent	C	1	v9	Number of urgent packets
Content features within a connection suggested by the knowledge domain				
hot	C	2	v10	The count of access to system directories, creation and execution of programs
num_failed_logins	C	2	v11	Number of failed login attempts
logged_in	D	2	v12	1 - successfully logged in; 0 - otherwise
num_compromised	C	2	v13	Number of "compromised" conditions
root_shell	C	2	v14	1 - root shell is obtained; 0 - otherwise
su_attempted	C	2	v15	1 – 'su root' command attempted; 0 - otherwise
num_root	C	2	v16	number of 'root' accesses
num_file_creation	C	2	v17	Number of file creation operations
num_shells	C	2	v18	Number of shell prompts
num_access_files	C	2	v19	Number of writes, delete and create operations on access control files
num_outbound_cmds	C	2	v20	Number of outbound commands in an ftp session
is_hot_login	D	2	v21	1- the login belongs to the 'hot' list (e.g., root, adm, etc.); 0 – otherwise

is_guest_login	D	2	v22	1 - the login is a ‘guest’ login (e.g., guest, anonymous, etc.); 0 – otherwise
Traffic features computed using a two-second time window				
count	C	3	v23	Number of connections to the same host as the current connection in the past 2 seconds
srv_count	C	3	v24	Number of connections to the same service as the current connection in the past 2 seconds
serror_rate	C	3	v25	% of connections that have ‘SYN’ errors to the same host
srv_serror_rate	C	3	v26	% of connections that have ‘SYN’ errors to the same service
rerror_rate	C	3	v27	% of connections that have ‘REJ’ errors to the same host
srv_rerror_rate	C	3	v28	% of connections that have ‘REJ’ errors to the same service
same_srv_rate	C	3	v29	% of connections to the same service and the same host

diff_srv_rate	C	3	v30	% of connections to different services and the same host
srv_dif_host_rate	C	3	v31	% of connections to the same service and different hosts
Host-based features				
dst_host_count	C	3	v32	Number of connections to the same host to the destination host as the current connection in the past 2 seconds
dst_host_srv_count	C	3	v33	Number of connections from the same service to the destination host as the current connection in the past 2 seconds
dst_host_same_srv_rate	C	3	v34	% of connections from the same service to the destination host
dst_host_diff_srv_rate	C	3	v35	% of connections from the different services to the destination host
dst_host_same_src_port_rate	C	3	v36	% of connections from the port services to the destination host
dst_host_srv_diff_host_rate	C	3	v37	% of connections from the different hosts from

				the same service to a destination host
dst_host_serror_rate	C	3	v38	% of connections that have 'SYN' errors to the same host to the destination host
dst_host_srv_serror_rate	C	3	v39	% of connections that have 'SYN' errors from the same service to the destination host
dst_host_rerror_rate	C	3	v40	% of connections that have 'REJ' errors from the same host to the destination host
dst_host_srv_rerrot_rate	C	3	v41	% of connections that have 'REJ' errors from the same service to the destination host

Abb. *C- Continuous, D- Discrete, 1- Intrinsic, 2- Content, 3- Traffic
Source: https://kdd.ics.uci.edu/databases/kddcup99/task.html, (Mukkamala and Sung, 2003)

Moreover, the categorical features attached to protocol_type, service, and flag have different values and the same have been mentioned below in Table-1.7 including a detailed description of the flag value separately in Table-1.8.

Table-1.7: Different values of protocol, service, and flag

Protocol_type (v2)	Label	**Service v3**	Label	**Service v3**	Label	**Service v3**	Label
tcp	1	aol	1	http_8001	25	red_i	49
udp	2	auth	2	imap4	26	remote_job	50
icmp	3	bgp	3	irc	27	rje	51
Flag V3		courier	4	iso_tsap	28	shell	52
OTH	1	csnet_ns	5	klogin	29	smtp	53
REJ	2	ctf	6	kshell	30	sql_net	54
RSTO	3	daytime	7	ldap	31	ssh	55
RSTOS0	4	discard	8	Link	32	sunrpc	56
RSTR	5	domain	9	login	33	sundup	57
S0	6	domain_u	10	mtp	34	systat	58
S1	7	echo	11	name	35	telnet	59
S2	8	eco_i	12	netbios_dgm	36	tftp_u	60
S3	9	ecr_i	13	netbios_ns	37	tim_i	61
SF	10	efs	14	netbios_ssn	38	time	62
SH	11	exec	15	netstart	39	urh_i	63
		finger	16	nnsp	40	urp_i	64
		ftp	17	nntp	41	uucp	65
		ftp_data	18	ntp_u	42	uucp_path	66
		gopher	19	other	43	vmnet	67
		harvest	20	pm_dump	44	whois	68
		hostnames	21	pop_2	45	X11	69
		http	22	pop_3	46	Z39.50	70
		http_2784	23	printer	47		
		http_443	24	private	48		

Table-1.8: Detail description of the flag value

Flag	**Description**
RSTOS0	Originator sent an SYN followed by an RST, never see an SYN-ACK from the responder
RSTR	Established, responder aborted
RSTO	Connection established; originator aborted (sent a RST)
OTH	No SYN saw, just midstream traffic (a “partial connection” that was not later closed)
REJ	Connection attempt rejected
S0	Connection attempt seen, no reply
S1	Connection established, not terminated
S2	Connection established and a close attempt by originator seen (but no reply from responder)
S3	Connection established and a close attempt by responder seen (but no reply from originator)
SF	SF Normal establishment and termination
SH	Originator sent an SYN followed by a FIN (finish ‘flag’), never saw an SYN-ACK from the responder (hence the connection was “half” open)

1.12 NSL-KDD dataset Description

The NSL-KDD dataset is then processed with an advanced version of the KDD cup’99 dataset. Extensive researches have been performed by many researchers on the NSL-KDD dataset using different tools and techniques to attend a common objective in developing an Intrusion Detection System. To cite a few of them, the NSL-KDD dataset was analyzed using various machine learning techniques on WEKA. K-means clustering algorithms use the NSL-KDD dataset to train and test existing and new attacks and also to find accuracy. NSL-KDD dataset was compared with its counterpart KDD cup 99 dataset using SOM Artificial Neural Network. The NSL-KDD dataset was also analyzed in parallel with the KDD Cup’99 cup dataset using different data mining-based machine learning algorithms such as, Decision tree, K-means clustering algorithms, Support Vector Machine, etc. (Revathi and Malathi, 2013 and Dhanabal and Shantharajah, 2015) which gave a very productive and accurate result.

The other advantages of the NSL-KDD dataset over the KDD dataset are as follows.

(i) No redundant records in the train set, so the classifier will not produce any biased results.

(ii) No duplicate record in the test set which has better reduction rates.

(iii) The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD dataset.

The training dataset is made up of 21 different attacks out of the 37 presents in the test dataset. The known attack types are those present in the training dataset while the novel attacks are the additional attacks in the test dataset i.e. not available in the training dataset. The attack types are grouped into four categories: DoS, Probe, U2R, and R2L.

1.13 Statement of the Problem

The state-of-the-art Intrusion Detection Systems in Artificial Neural Network and Support Vector Machine have posted the following problems which promoted the researcher for research.

(i) Multiple issues are connected to intrusion detection in Artificial Neural Network and Support Vector Machine.

(ii) The present Intrusion Detection Systems do not cover the complete detection coverage in both Artificial Neural Network and Support Vector Machine.

(iii) Improvement in detection performance of the available Intrusion Detection Systems without affecting the false alarm rate.

(iv) Lacking a proper model for an attack-detector relationship in both Artificial Neural Network and Support Vector Machine.

(v) Anomaly detection is an important problem in the dynamic network domain.

1.14 Scope of the Study

Research in the intrusion detection system is an emerging area in computer science and network security. The increasing volume of network traffic and unauthorized users into the network makes the computer network more vulnerable. To deal with the increasing network traffic and new kind of attacks, more research on Intrusion Detection Systems and specifically NIDS are very much important. An everyday computer network is experiencing a different kind of traffic, therefore to protect our data while transmitting system need regularly update. The scope of the present study is limited to the applications of ANN and SVM for classification in network intrusion detection using both KDD Cup’99 dataset and NSL-KDD dataset. The experiments of both Artificial Neural Network and Support Vector Machine are carried out and the performances are compared by using various performance matrices. Here, both MATLAB and KEEL software are used for determining the evaluation results and compare the performance.

The proposed study focuses on a comparative study between two data mining techniques i.e, Artificial Neural Network (ANN) and Support Vector Machine (SVM). For this purpose, network traffic data are required for training and testing. Research reveals that the popularly used KDD’99 dataset carries some redundant sets of records which may cause biases in the result. For the present research work both NSL-KDD, an updated or modified version of the KDD’99 dataset, and original KDD’99 is used. A huge amount of dataset is required to store and represent properly so that the analysis and comparison become easier.

1.15 Review of Literature

Kabir *et al.* (2017), proposed a novel approach for intrusion detection systems based on sampling with Least Square Support Vector Machine (LS-SVM). Decision-making is performed in two stages i.e, first, they divided the whole dataset into some predetermined arbitrary subgroups, and in the second phase applied the Least Square Support Vector Machine (LS-SVM) to the extracted samples to detect intrusions. The proposed algorithm as optimum allocation-based Least Square Support Vector Machine (OA-LS-SVM) for Intrusion Detection System. They performed experiments on the KDD'99 database. The authors tested all binary-classes and multiclass to obtain a realistic performance in terms of accuracy and efficiency.

Rout *et al.* (2017), observed that real-world application encounters the problems of class imbalance as the presence of the imbalanced dataset impedes the performance of the standard learning algorithms. They viewed that, it is difficult to handle the multi-class imbalance problem than the binary class imbalance problem. They discussed different techniques to accept the challenges of the multi-class imbalanced dataset. They used different kinds of the multi-class imbalanced dataset and five types of Boosting methods and experimented with the KEEL repository.

Belavagi and Muniyal (2016) built new classification and predictive models for intrusion detection by using different machine learning classification algorithms such as Logistic Regression, Gaussian Naive Bayes, Support Vector Machine, and Random Forest. The authors tested the algorithms with the NSL-KDD dataset and found that Random Forest Classifier outperforms the other methods for identifying whether the data traffic is normal or an attack.

Bamakan *et al.* (2016), introduced the time-varying inertia weight, penalized MCLP to deal with unbalanced dataset and acceleration coefficients to CPSO. The authors applied both feature selection and parameter setting techniques simultaneously to MCLP and SVM. They also proposed a weighted objective function to evaluate the proposed IDS framework. They also proposed an IDS framework that obtained a low false alarm rate and a high detection rate. The performance of the proposed methods was evaluated by conducting experiments with the NSL-KDD dataset, which was derived and modified from well-known KDD cup'99 dataset. The results found that the proposed method performed better in terms of having a high detection rate and a low false alarm rate when compared with the obtained results using all features.

Folino and Sabatino (2016) presented the current state of the art of the ensemble-based methods used in modern intrusion detection systems concerning distributed approaches and implementations. They viewed that, designing appropriate NIDSs requires sharing knowledge across multiple nodes. They discussed some open issues and lessons and suggested designing more efficient NIDSs.

Inayat *et al.* (2016), presented an IRS taxonomy based on design parameters to classify existing schemes and investigated the essential response design parameters for IRS to mitigate attacks in real-time and obtain a robust output. The authors discussed comprehensively the design parameters and qualitatively analyzed existing IRS schemes based on the response design parameters. They identified open research challenges to highlight key research areas.

Aggarwal and Sharma (2015) viewed that, the KDD dataset is a well-known benchmark in the research of Intrusion Detection techniques. They confined their discussion of the analysis of the KDD dataset concerning Basic, Content, Traffic, and Host.They performed the analysis concerning two prominent evaluation metrics, Detection Rate (DR) and False Alarm Rate (FAR) for an Intrusion Detection System (IDS).

Al-mamory and Jassim (2015) suggested two grains levels intrusion detection system (IDS) i.e, fine-grained and coarse-grained. They found that the most suitable IDS level is the coarse-grained to increase IDS performance. They visualized that, the moment any intrusion is detected by coarse-grained IDS, the fine-grained is activated to detect the possible attack details. The authors used a fast decision tree algorithm in both of these detection levels and tested the model on KDD CUP'99 offline dataset and a real traffic dataset. The experimental results demonstrated that the proposed model is highly successful in detecting known and unknown attacks and can be successfully adapted with packets' flow to increase IDS performance.

Duque and Omar (2015) in their paper discussed the problem shared by current IDS which is the high false positives and low detection rate. They used k-means for unsupervised machine learning and proposed a model for Intrusion Detection System (IDS) which is having a higher efficiency rate & low false positives and false negatives. They used the NSL-KDD dataset with 25,192 entries with 22 different types of data. Their work found the best results when the number of clusters matches the number of data types in the dataset. They recommended for k-means data mining algorithm followed by a signature-based approach to lessen the false-negative rate.

De-la-hoz *et al.* (2015), discussed the growth of the Internet and the number of interconnected computers which exposed significant amounts of information to intruders and attackers. The authors proposed different detection approaches including the use of machine learning techniques based on neural models such as Self-Organizing Maps (SOMs). They presented a new classification approach that hybridizes both statistical techniques and SOM for network anomaly detection.

Jabez and Muthukumar (2015) discussed that the Intrusion detection and prevention systems (IDPS) identify the possible incidents, logging information about them, and report attempts. They viewed that, IDPS is essential to the security infrastructure of any organization. They proposed a new approach called outlier detection where the anomaly dataset is measured by the Neighbourhood Outlier Factor

(NOF). The experimental results proved that the proposed approach identifies anomalies very effectively than any other approach.

Ranshous *et al.* (2014), have viewed that anomaly detection is an important problem in the dynamic network domain. They focused their views through a current survey in time-evolving networks and have obtained data through five different categories based on different technical approaches consisting of nodes, edges, sub-graphs, and events.

Rejchrt (2014) concentred upon the anomaly survey with regards to its learning and understanding about network anomaly and viewed that it belongs to the network security community.

Sayer *et al.* (2014), visualized that, there is a threat of being robbed and there is a possibility of several numerable attacks on the computer due to an increase in the speed of information data flow.

Shrivastava *et al.* (2013) observed that the network threats and security raised a major issue with regards to the data integrity and loss of data. They further expressed that, in the beginning, intrusion detection system performed on the process of the satirical frequency of audit system logs which, however, was applied subsequently in many directions in the network which include, data mining techniques, neural network, expert system and soft computing approach such as, fuzzy logic, genetic algorithm, and machine learning, etc.

Wang and Yu (2013) presented an intrusion detection system of a hybrid neural network model based on the Radial Basis Function (RBF) network and Elman network. Mention may be made that, RBF network uses a local index attenuation nonlinear function to do the local approximation works for nonlinear input/output while the Elman network is used to memorize the previous events. Further, the RBF network has a fast convergence speed of learning (Wang and Yu, 2013) compared with MLP. According to them, there are a good number of studies in this area, and the widely applied network is a kind of multilayer feedforward neural network, but the MLP network model does not have memory function of previous events, and MLP network needs long training time network. The authors further viewed that, it is used for anomaly detection and misuse detection. This model has a memory function. It can detect discrete and related aggressive behaviour effectively. RBF network is a real-time pattern classifier, and the Elman network achieves the memory ability for the former event. Based on the hybrid model, the intrusion detection system uses the DARPA dataset to do test evaluation. It uses a ROC curve to display the test result intuitively. After the experiment, it proves this hybrid model intrusion detection system can effectively improve the detection rate, and reduce the rate of false alarm and failure.

Panko (2012) mentioned the use in Intrusion Detection System filtering mechanisms, Application Specific Integrated Circuits for processing power, Attack Confidence Identification Spectrum along with possible actions like Drop Packets, Bandwidth Limitation for certain types of traffic, etc.

Singh *et al.* (2012), observed that an intruder continuously monitors the network and host activities for detecting attacks into the network and the task of intrusion-detection has also monitored the usage of such systems and detects the apparition of insecure states.

Vamsidhar *et al.* (2012), observed that web-based applications in multiple dimensions have resulted to increase problems on security. They suggested developing preventive measure mechanisms by dividing the attacks into several groups in the system so as increase the efficiency of detecting the unknown attacks.

Vinchurkar and Reshamwala (2012) while discussing the IDS pointed out that, it is essential for network security and intrusion attacks. They further visualized that, monitoring activities of the network and threats as well are the essential features of IDS which can be classified as Data and Model of intrusion, and suggested for Support Vector Machine use to specify the classifier construction problem.

Zhao and Li (2012) proposed a new model to enhance the classification effectiveness of the Support Vector Machine (SVM). They viewed that, penalty parameter ***c*** and kernel function parameter ***g*** of SVM are optimized using the genetic algorithm of binary coding, and the optimized model GA-SVM is established. The dimensionality of the input sample for SVM is reduced by PCA (Principal Component Analysis) and the model GA-PCA-SVM is established. They used the KDD Cup 1999 dataset to evaluate the proposed model and carried out an experiment based on GA-SVM and GA-PCA-SVM. They deduced that the genetic algorithm optimization improves the classification accuracy rate of SVM and PCA operation shorts the training time and test time.

Somer (2010), categorically discussed the NIDS which is a major security component in network environments. The author viewed that, there is a gap between claims and operational reality. The author of this book emphasized Bro-NIDS which is a powerful and flexible open-source research system running on commodity hardware.

Wang *et al.* (2010), pointed out that, researchers argue for the implementation of Artificial Neural Networks (ANNs) to improve the IDS performance of Intrusion Detection systems (IDS) compared to other traditional methods. The authors further discussed that, for ANN-based IDS, detection precision, especially for low-frequent attacks, and detection stability are still required to be enhanced. They proposed a new approach, called FC-ANN based on ANN and fuzzy clustering to solve the problem and help IDS for achieving a higher detection rate, less false positive rate, and stronger stability. They mentioned following the general procedure of FC-ANN where fuzzy clustering technique is the first step that is used to generate different training subsets followed by the second step based on different training subsets, different Artificial Neural Network models to be trained to formulate different base models and the third and final step involves a meta-learner, fuzzy aggregation module which is employed to aggregate these results. Experimental results on the KDD CUP 1999 dataset showed that the proposed new approach, FC-ANN, outperforms BPNN and other well-known methods such as decision tree, the naive Bayes in terms of detection precision and detection stability.

Cherkasova *et al.* (2009), discussed the importance of automated tools for understanding application behavior including its need for performance analysis and debugging tasks. The authors further proposed a novel framework for automated anomaly detection and application change analysis.

Schuster (2008), discussed the robustness of Artificial Neural Networks (ANNs) and proposed several novels, nature-inspired ANN architectures. He viewed that, a robust system is a system that tolerates faults and it is recognized as a ubiquitous feature in many systems. Scientists are inclined to study in this area. While applying robust in traditional ANN architecture, the author further opined that Artificial Neural Network constitutes with one input layer, one hidden layer, and one output layer and the network has five input neurons (attributes *a*1 to *a*4, plus a bias), six neurons in the hidden layer, and three output neurons (*o*1 to *o*3).

Shum and Malki (2008) presented a neural network-based intrusion detection method for Internet-based attacks on a computer network. According to the author, while Intrusion detection systems (IDS) are concerned with predicting and thwart current and future attacks, the neural networks identify and predict unusual activities in the system. They applied feedforward neural networks with the backpropagation training algorithm in the study.

The Technical Report submitted by Chandola *et al.* (2007), discussed Anomaly detection, an emerging research area and this is a template that provides an easier and succinct understanding of the techniques belonging to each category.

Srinoy *et al.* (2007), viewed that one main drawback of the intrusion detection system is the inability of detecting new attacks that do not have known signatures. A discussion on the computational complexity of the techniques also has been formulated since it is an important issue in real application domains.

Robertson *et al.* (2006), while discussing anomaly web attacks using generalization and characterization techniques viewed that, the custom and ad-hoc nature of web applications makes learning-based anomaly detection systems a suitable approach to provide early warning about the exploitation of novel vulnerabilities. They observed anomaly-based systems contribute to producing a large number of false positives along with providing poor or non-existent information about the type of attack that is associated with an anomaly. The paper explained the novel approach to anomaly-based detection of web-based attacks which uses an anomaly generalization technique that automatically translates suspicious web requests into anomaly signatures which are used to group recurrent or similar anomalous requests to facilitate an administrator to deal with such alerts. They further viewed that, a heuristics-based technique is used to infer the type of attacks that generate the anomalies. This enables the prioritization of the attacks and provides better information to the administrator.

Chen *et al.* (2005), viewed that, while Support Vector Machine with $tf{\times}idf$ scheme achieved the best performances ANN with a simple frequency-based scheme achieved the worst.

Douligeris and Mitrokotsa (2004), discussed that Denial of Service (DoS) attacks constitute one of the major threats and among the difficult security problems in the present Internet age and major concern lies with Distributed Denial of Service (DDoS) attacks and its impact is terrible. They further pointed out that, a DDoS may attack can easily exhaust the computing and communication resources of its victim within a short time with or without any alarm. Because of the seriousness of the problem many defense mechanisms have been proposed to combat these attacks. The authors presented a structural approach to the DDoS problem by developing a classification of DDoS attacks and DDoS defense mechanisms. Furthermore, the important features of each attack and defense system category are described and the advantages and disadvantages of each proposed. The primary objective of the paper is to place some order into the existing attack and defense mechanisms, to facilitate a better understanding of DDoS attacks and develop techniques and procedures to combat these attacks that may be developed through effective algorithms.

Moradi and Zulkernine (2004) discussed the soft computing-based methods and presented a neural network approach to intrusion detection. They used a Multi-Layer Perceptron i.e. part of an artificial neural network that returns 0 or 1 according to the value of a linear function of its inputs analysis approach.

Mukkamala *et al.* (2003), projected intrusion detection and audit trail reduction using Support Vector Machine (SVMs) and neural networks. They opined that efficient and highly accurate classifiers could be built using either Support Vector Machine or Neural Networks for intrusion detection. According to them, both Support Vector Machine and Neural Networks deliver highly accurate i.e. 99% and higher performance. Better results could be derived using Support Vector Machine.

Mukkamala and Sung (2003) focussed on the use of artificial intelligence techniques for offline intrusion analysis and to protect the integrity and confidentiality of the information infrastructure. They opined that an effective forensic tool is essential for ensuring information assurance by updating the newly identified security breaches into the organization's protection and detection mechanisms.

This section reviews the research particularly about the use of Support Vector Machine for network intrusion detection, specifically as applied to the KDD cup 1999 dataset. Early research on SVMs applied to intrusion detection was performed by Mukkamala *et al*. (2002), where neural networks were compared with Support Vector Machine. SVMs were chosen because of noticeable advantages over neural networks in terms of speed and scalability and the researchers utilized a radial basis function kernel. In their study, the authors found that both Artificial Neural Networks and Support Vector Machine had high accuracy, but the SVMs could only classify the intrusion dataset into two classes, i.e.“attack” or “normal”. In this study, only very small dataset were used (7312 data points for the training dataset and 6980 in the testing set). The researchers claimed that, although SVMs and NNs have been successfully applied for the network intrusion detection problem in the past, these techniques have tended to take a long time to train the intrusion dataset and required complex parameter tuning. Moreover, SVMs don’t have built-in multiclass classification capabilities.

Cheng and colleagues (2012) used the KDD CUP 99 for their research work on small-sized dataset of 2000, 4000, and 8000 samples. They compared the performance between MATLAB implementation of both types of ELMs and a C language implementation of a Support Vector Machine using the LIBSVM library (Chang and Lin, 2011).

The Support Vector Machine was adapted as the classification technique to handle multiclass classification problems using a max-win voting method. Finally, the researchers concluded that both types of ELMs ran faster than SVM but the basic ELM had “slightly lower accuracy” than Support Vector Machine while Kernel-based ELM’s accuracy was similar to that of SVM. In their experiments, the authors noted that in the multiclass classification problem, the Support Vector Machine took a much longer time.

1.16 Objectives of the Study

The objectives of the present study are to,

(i) Study of two existing networks i.e, Artificial Neural Network (ANN) and Support Vector Machine (SVM) in Intrusion Detection System.

(ii) Analysis of both KDD’99 dataset and NSL-KDD dataset and pre-processing.

(iii) Carry out experiments using different algorithms of Artificial Neural Network (supervised) and various classification techniques of Support Vector Machine (supervised) and compare the performance.

1.17 Methodology

For the present research work, the scholar studied the two existing networks i.e., (i) Artificial Neural Network (ANN) and (ii) Support Vector Machine (SVM) for Intrusion Detection System. The scholar obtained both the KDD and NSL-KDD dataset (a modified version of the KDD CUP’99 dataset) for all the experiments, from DARPA Intrusion Detection Evaluation Program at MIT's Lincoln Laboratory to analyze for pre-processing. The pre-processed data were used for both Artificial Neural Network and Support Vector Machine to carry out different experimental results and compared the performance.

1.18 Chapterization

The present study is divided into six chapters. Chapter-1 of the study discusses Introduction, Need of security, Detection, Prevention, Recovery, Concept of the computer security, Confidentiality, Data Confidentiality, Privacy, Integrity, Data Integrity, System Integrity, Availability, Authenticity, Accountability, Security Attacks, Passive Attacks, Release of Message Contents, Traffic Analysis, Active Attacks, Masquerade, Replay, Modification of Messages, Distributed Denial of Service (DDoS), Denial of Service (Denial of Service Attack), Distributed Attack, Insider Attack, Close-in-Attack, Phishing Attack, Hijack Attack, Spoof Attack, Buffer Overflow, Exploit Attack, Password Attack, Security Mechanism, Specific, Security Mechanism, Pervasive

Security Mechanism, Principles of Security, Reconnaissance, Exploitation, Reinforcement, Consolidation, Server Consolidation, Storage Consolidation, Pillage, Threat, Division of Threats, Unauthorised Users Access Role Accounts, Authorised Users Accessing Role Accounts, Security Threat, Data Distribution in KDD’99 dataset and its features, NSL-KDD dataset Description, Statement of the Problem, Scope of the Study, Survey of Literature, Objectives of the Study and Methodology.

Chapter-2 of the study elaborately discusses the theoretical application of Artificial Neural Network (ANN) and Support Vector Machine (SVM) in the Intrusion Detection System. The chapter also includes discussions on Growth of Internet, Internet Attack- A Genealogy, Authentication, Access Control, Non-repudiation, Security Services (X.800) and RFC 2828, Network Security, Network Security Issues, Network Security Attacks.

The chapter also discusses Intrusion Detection System- The Notion, External Penetrator, Masquerader, Misfeasor, Clandestine User, E-Modules, D-Modules, A-Modules, R-Modules, Need of Intrusion Detection System, Taxonomy of Attacks and Intrusions, Intrusion Detection System Approaches, Preemptive Blocking, Infiltration, Trojan horse, Worms, Virus, Hoax, Intrusion Deflection, Intrusion Deterrence, Statistical Anomaly Detection, Audit Records, Architecture of Intrusion Detection System, Information (Data) Source, Types of Intrusion Detection Systems, Host-based Detection System (HIDS), Network-Based Intrusion Detection System (NIDS), Vulnerability Assessment Intrusion Detection System (VAIDS), Distributed Intrusion Detection System (DIDS) including Approached Based Intrusion Detection System, Intrusion Prevention System (IPS), Neural Network in Intrusion Detection System, Applications of Artificial Neural Network, Learning Process, Support Vector Machine in Intrusion Detection System, Application of Support Vector Machine, advantages and disadvantages, etc. This chapter also includes the experimental setup (KEEL), the experiment of the KDD (training and test) dataset, and the performance matrices which are used to measure the performance of all the algorithms.

Chapter-3 discusses various Artificial Neural Network algorithms used for experiments based on NSL-KDD dataset. The algorithms include LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN. The chapter also includes a brief description of the experimental setup (KEEL), description of the NSL-KDD dataset, proposed experimental framework. The performance matrices used to measure the performance of all the algorithms are also discussed in the chapter.

Chapter-4 describes detailed experiments using various Support Vector Machine classification techniques. It also includes different kernel tricks and functions, application of SVMs for both classification and regression including the experimental setup (MATLAB), description of the KDD and NSL-KDD dataset, proposed experimental framework to carry out the experiments. The chapter also includes different performance matrices that were used to measure the performances.

Chapter-5 of the study focuses on the performance comparison according to the framework for both Artificial Neural Network (ANN) and Support Vector Machine (SVM) including experimental findings, Online Algorithm for Lagrangian Support Vector Machine, Robust Online learning algorithm with Lagrangian Support Vector Machine (ROLALSVM), Performance Evaluation and Chapter- 6 discusses the chapter wise summary and conclusions followed by directions on the prospective areas for future work. The work concludes with a comprehensive bibliography.

CHAPTER-2

APPLICATION OF ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINE IN INTRUSION DETECTION SYSTEM

Chapter-2

Application of Artificial Neural Network and Support Vector Machine in Intrusion Detection System

2.1 Introduction

The adaptability of the Internet in every operation has revolutionized the information world as well as trade and commerce, industries, education, insurances, etc. Since 1970, it has been illuminated as a visionary of connections by way of connecting millions and millions of people in the virtual domain which otherwise according to McLuhan (1987) could be termed as Global Village. It is being the global platform that has been recognized as the most powerful and dominating podium in the communication domain and it achieved significantly the goals and objectives. In the process of its accessibility into multifarious operations, intrusion in any form cannot be ruled out.

The proliferated use of the internet precipitated the growth of attackers in a parallel who gain unauthorized access to system services, resources, and information or make effort to compromise with system integrity. An attack can be grouped into two types such as (i) Active Attack and (ii) Passive Attack. While the active attack relates to an attempt to modify the system resources or affect the operation, the passive attack connotes an attempt to learn or use the information from the system without affecting the system resources (Shirey, 2000). However, the other prevailing types of attacks such as Distributed Attack, Close-in-Attack, Phishing Attack, Hijack Attack, Spoof Attack, Buffer Overflow, Exploit Attack, etc. have been discussed in the foregoing chapter. Internet Security Glossary (Shirey, 2000) defines an attack as an assault on system security that derives from an intelligent threat. It is intelligent behaviour to avoid security policy deliberately and such an act is identified as penetration, violation, vulnerability, etc. Further, an attack is also committed in two ways i.e, (i) inside attack which precipitates from the organization itself, and (ii) outside attack that hails from the outside the organization. Shirey (2000) defined the inside attack as an attack originated by an entity inside the security margin i.e, an insider who is authorized to access the system resources but performs illegitimately. An outside attack initiates from outside the boundary through an unlawful way on the internet to the system. In any case, the attackers ramp up their attempts with a violent attitude to defunct the system (Anwar *et al.*, 2017).

2.2 Growth of Internet

The Internet has become global reach and integrity in terms of support services and applications which require some basic agreements and social behaviour between technologies and human beings even if there is no central authority that designates or permits different classes of internet activities. The technological developments supplemented with the exorbitant use and trust upon the global podium i.e, the Internet, extended limitless new opportunities to facilitate personal, professional information, experiences, and multifarious activities such as finance, industry, trade, and commerce, education, etc. However, in tune with the growth of technology, these improvements are also being widely developed as a tool or infrastructure for committing many criminal offenses. Multidimensional types of crimes are being committed daily on the Internet and these are amplified through the Internet that directly targets compromising devices or trick victims into enabling vulnerable features or exposing user credentials and other sensitive information. The growing Internet penetration and the newly emerging technologies such as the Internet of things are also changing people’s behaviour online and create a broader attack surface, new attack vectors, and more points of entry, such as through social engineering techniques, which was also a key finding of Europol’s Internet Organised Crime Threat Assessment (IOCTA) 2014.

2.2.1 Internet Attack- A Genealogy

While tracing a genealogy of attack on the Internet it could be discussed that the concept of a worn program that spreads from machine to machine was first conceived by John Brunner in 1975 and termed the program as tapeworms that lived "inside" the computers. During, 1979-1981, researchers at Xerox PARC developed and experimented with worm programs that work in a distributed environment. Further, the notion of a ‘morris worm’, a standalone malware computer program was first conceived on 2nd November 1988 by R.T. Morris in the USA, which primarily was intended to collect host, network, and user information and then broke into other machines using flaws present in those systems’ software (Spafford, 1988). In a subsequent development, new attacks such as Denial of Service (DoS) during the 1990s, Distributed Denial of Service (DDoS) in 1999, Botnets, storm Botnets, Blended attacks in 2001, Information warfare, were developed. Blended attacks with the release of Code Red Worm followed by Nimda, Slammer, Blaster worms are the potential in computer and network attacks.

The persistent use of the Internet for various purposes as discussed above precipitated computer and networks attack which gained momentum over the period. The attackers equally not only float in a parallel way as that of internet users in the global sphere but also, mounting abruptly including their strength and sophistication. Figure-2.1 placed below explains the Attack Sophistication vs. Intruder Technical Knowledge.

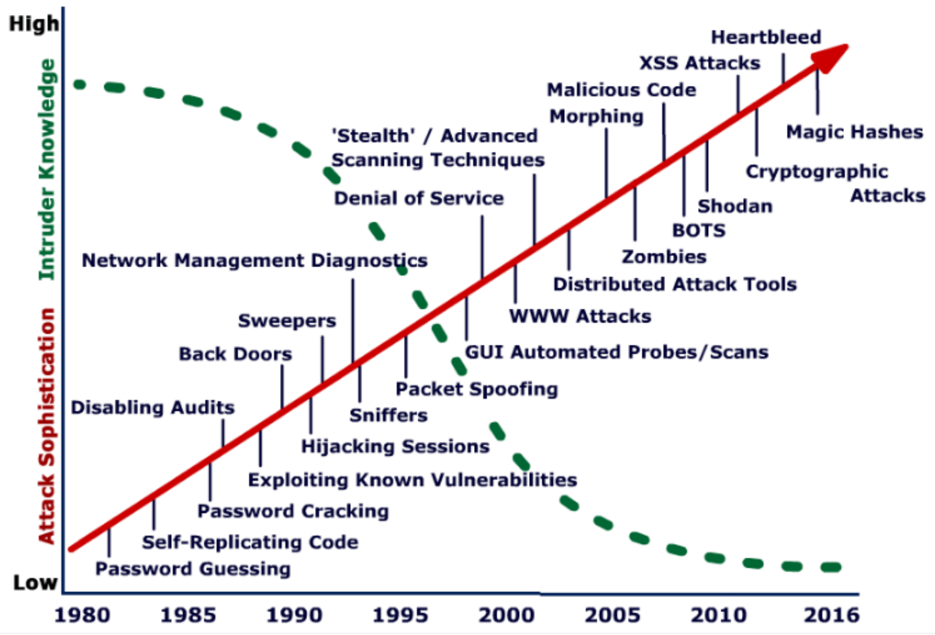


Fig 2.1: Attack Sophistication vs. Intruder Technical Knowledge
Source: Sorell (2016). Projects:2016s1-160a Cyber Security - IoT and CAN Bus Security (https://www.eleceng.adelaide.edu.au/)

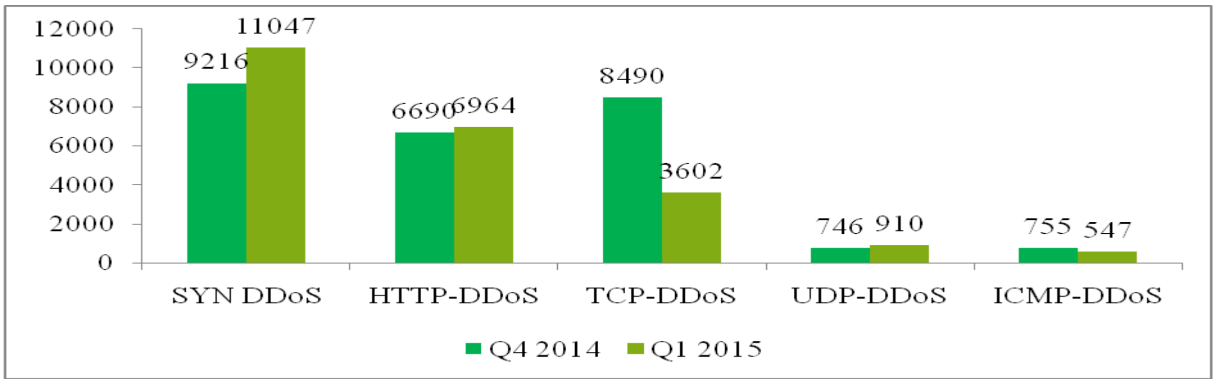
An assessment such as multifarious Internet attacks released by Kaspersky Lab happens to be long-standing recognized expertise in combating cyber threats, including DDoS attacks of different types and varying degrees of complexity. The data relates to DDoS Intelligence statistics from 1 October to 31 December 2014, i.e Q4 2014 and 1 January to 31 March 2015 i.e, Q1- 2015 placed in Table-2.1

supplemented with Graph-2.1 which after analysis revealed the following comparison in a global platform.

Table- 2.1: DDoS Intelligence statistics from 1 October to 31 December 2014

Attack	**Q4 2014**	**Q1 2015**
SYN DDoS	9216	11047
HTTP-DDoS	6690	6964
TCP-DDoS	8490	3602
UDP-DDoS	746	910
ICMP-DDoS	755	547

Source: Statistics on Botnet assisted DDoS Attacks Q1 2015, https:// securelist. com/ blog/research/70071/statistics-on-botnet-assisted-ddos-attacks-in-q1-2015 https://securelist.com/author/kaspersky/



Graph- 2.1: DDoS Intelligence statistics from 1 October to 31 December 2014, i.e, Q4 2014 and 1 January to 31 March 2015 i.e, Q1- 2015.

2.3 Network Security- Issues

Network security is concerned with extending freedom from any type of threat or danger in the network, computer system, and the resources as well as the company, organization or network administrator, or individual. It further extends protection from any unauthorized access to the malicious components as well as monitoring consistently and measure the effectiveness or lack of effectiveness of the network. However, it is a major concern for every organization, institution, information center, business setup, financial transaction while performing multifarious activities on the network platform. A major concern in network security is that a competitor or any hacker can gain access to sensitive or critical data and may even delete or make off with the information resulting in data loss or complete system destruction.

Both information security and network security are interchangeably used most of the time to represent the same concept. However, network security more specifically is used for protection by outside intruders. Further, network security is a major component of information security. Information security which is a wider dimension includes security issues relating to security policies, security auditing, security assessment, trusted operating system, database security, secure code, emergency response, computer forensics, software forensics, disaster recovery, and security training.

The various issues in network security are explained below.

- **Authentication**

It is concerned with security measures designed specially to institute the validity of transmission information, message or originator, or a way to confirm the authorization of an individual to receive the information.

- **Access Control**

In network security, access control is associated with the ability to restrict access to host systems and applications through communication links. Proper identification/ authentication is required for access to the system.

- **Non-repudiation**

It precludes either the sender or receiver from refusing a transmitted message. A non-repudiation service affords assurance of the beginning or delivery of data/message to defend the sender in opposition to false denial by the recipient that the information has been obtained or to shield the recipient against false denial with the aid of the sender that the information has been sent. Thus, a non-repudiation carrier delivers evidence to stop unilaterally enhancing or terminating criminal obligations springing up out of a transaction effected by computer-based capability (Zhou and Gollmann, 1997).

- **Security Services (X.800) and RFC 2828**

Security service can be defined as a capability that supports one, or any of the security objectives which comprise confidentiality, integrity, and availability. While X.800 explains that security service is provided by a protocol layer of communicating open systems and it assures enough security of the systems or data transfer, RFC 2828 specifies it as processing or communication service which is provided by a system to keep intact to the system resources (Stallings and Brown, 2010).

- **Security Mechanism**

Security mechanism implies a machine designed to grant one or more security services commonly rated in phrases of the energy of provider and assurance of the design. This can be defined as a method or technique for enforcing a safety policy. Mechanism denotes the sense of non-technical in its function as it requires proof of identity before altering a password. However, the policy requires some procedural mechanisms that technology cannot impose (Bishop and Venkatramanayya, 2005). X.800 while defining security mechanism has emphasized its application in (i) specific protocol layers e.g. Transmission Control Protocol (TCP) or an application layer protocol and (ii) unspecific protocol layer or security service.

The security mechanism as recommended by X.800 can be categorized into two types such as, (i) Specific Security Mechanism, and (ii) Pervasive Security Mechanism (Stallings, 2011). The Specific Security Mechanism includes (i) Encipherment, (ii) Digital Signature, (iii) Access Control, (iv) Data Integrity, (v) Authentication Exchange, (vi) Traffic Padding, (vii) Routing Control, and (viii) Notarization. The Pervasive Security Mechanism comprises (i) Trusted Functionality and, (ii) Security Label (Integrity and Sensitivity).

Thus, network security has become crucial in the sound functioning of the computer on the Internet and the primary objectives rest on providing the freedom to the users to exercise their rights and interests intrepidly (Wang, 2009 and Rowton, 2005). The network security is based not only on different layers of protection of the system but also on multiple components, networking monitoring, and security software, and hardware. The common threats encountered in the network domain consist of,

- Viruses, worms, and Trojan horses;
- Spyware and adware;
- Zero-day attacks, also called zero-hour attacks;
- Hacker attacks;
- Denial of service attacks;
- Data interception and theft; and
- Identity theft.

2.4 Network Security Attacks

A computer network attack signifies an action undertaken while using the computer in a network environment and it is a set of malicious activity that disrupts, denies, degrades, and or destroys information available in the computers and network-based computers. (Kissel, 2013). A network attack is executed through the data stream on the networks and aims to compromise the Integrity, Confidentiality, or Availability of the computer network system.

2.5 Security Model

The basic security model consists of four components such as (i) Cryptosystem, (ii) Firewalls, (iii) Anti-malicious software system, and (iv) Intrusion Detection System (Denning, 1987). The cryptosystem happens to be a security model that uses both computer cryptography and security protocol to protect data. The security protocols include encryption protocol also known as a cryptographic protocol, authentication protocol, and key-management protocols. A firewall is another component in the network security that verifies and authenticates the users to access the programs or services in the network. It also has an impact on the capacity to stop any unauthorized entry but, it fails to take a look at dangerous contents like computer worms that are transmitted across the network. The Anti-malicious software protects the systems from viruses, worms, Trojan horse, spyware that affect the file system, file formats, operating systems and it is a mechanism that can be employed for web security and denial-of-service defense (Wang and Kisssel, 2015).

2.6 Intrusion Detection System- The Notion

Anomaly pertains to an act, behaviour which is against the established, recognized, acceptable and normal behaviour. This otherwise can be explained as non-conforming patterns in different application domains. This is the consequential impact of the overwhelming growth of data in various areas such as Information Science, Medical Science, Engineering, and Computer Science is no exception to it. Rather, the penetration of computer technologies and other associated technologies has a positive brunt in precipitating anomalies. An anomaly is a malicious activity that could be traced in cyber-intrusion, terrorist activities, collapsing of a system and it has a common characteristic like disassociating from the normal behaviour or function (Chandola *et al.*, 2007). Thus, the anomaly can be likened to the ridiculous behaviour of a working system that causes serious threat and inconsistency and it is more prominent especially in a network system apart from other domains. In such a situation, the undesirable phenomenon needs instant intervention and proper understanding of the anomaly before collapsing the total system. Further, an intrusion detection system is an attempt at detecting intruders in a computer system or network. This is due to the proliferation of online networks in the World Wide Web domain and local area network.

The Internet, a global public network is well recognized as a viable and acceptable platform to take up various activities among the business communities to reach the end-users with their products, and simultaneously, they also encounter many unacceptable problems. Further, mention may be made that, it is a proliferated horizon in all sectors like banking, insurance, education, business, finance, communication, etc. and hence, it cannot be limited to one area. Further, the companies are operating their business globally through the web and hence, they are solely dependent upon networking. Therefore, the expanded horizon of business solely rests on the expeditious use of networks. In such circumstances, the intrusion has become a common element to make obstructions and this is recognised as an impediment to the growth and development of the system. Risks and vicious intrusions are growing at a parallel rate with the growth of sophisticated computing tools. Certainly, this activity is an anti-direction to the augmentation, especially in the economic zone. The highly connected computing world has also been equipped with intruders and hackers who could turn it damaged. This aggressive approach results in crippling the entire positivistic idea for sustainable development and thus, need security in manifold ways especially in a network environment to detect to take viable measures to restore the damages. Efforts from various sectors are penetrating the network domain to get rid of such an alarming situation. This mechanism is technically referred to as the Intrusion Detection System. (Barman and Khataniar n.d.).

Intrusion Detection System (IDS) involves a clustering mechanism not only to identify the illegitimate entry of intruders to the system which may cause-effect to the hardware, software, or both that controls the system in a network environment but also to detect the perpetrators before causing impairment to the data, resources, etc. (Vinchurkar and Reshamwala, 2012; Liao *et al.*, 2013). For smooth, faultless, consistency of system operations especially in a network domain, Intrusion Detection System has become an indispensable component as it not only defends to impairment of the system but also restrained from undesirable aggress, irrational, inconsistency behaviour of the computer. Added to these dimensions, it also allows a conducive environment by protecting the system from various operations in a network domain, audit network, and system configurations. Further, traffic monitoring and its analysis have become pragmatic not only on the Internet but also Intranet for determining the assorted problems and resolve the constraints through numerous tools.

Stefan Axelson (1999) has pointed out that, the concept of intrusion detection is analogous to the common burglar system to instrument a computer system or network for facilitating to detect the possible threat, violations to a security policy and raise an alarm to notify the proper authority that can be known as Site Security Officer (SSO). Anderson (Axelson, 1999) in earlier studies categorized the possible attackers in the computer system into four classes such as,

- **External Penetrator**

It comprises the employee of the organization, institutions that do not possess the right to gain access to resources that are otherwise known as illegitimate users. In a network domain, security is indispensable to protect the identity and authenticity of the resources.

- **Masquerader**

This is an attack where a fraudulent identity is used like network identity especially for gaining unlawful entry to personal computer data and information. It is being performed through legitimate access identification. There is a chance of extremely vulnerable to masquerade attack to a system if the authorization process is not completely defended from danger. Masquerade attacks can be committed not only through using pilfered logins and passwords but also by detecting gaps in computer programs, or tracing any means of penetration in the authentication process. In a public network domain, when the organization is connected, the attack is activated either by somebody inside the organization or by an unknown person outside the organization. The relative magnitude of access masquerade attackers gets contingent upon the level of authorization they have accomplished. As such, masquerade attackers can have a full assortment of cybercrime probabilities if they have derived the highest access authority to a business organization. Further, they also focus on personal attacks to harm the system though, the degree of harmfulness is less.

The masquerading attacker is triggered through vulnerable authentication because of its supportive activity to gain access by the attackers. In such a situation, once the attackers are successful in gaining access, the organizations' systems fall prey to them and the attackers start not only stealing the sensitive data but also manipulate and delete the critical data. They also take serious measures to alter the routing information including network configuration. There are multiple means of materializing such attacks. For example, the account of an authorized user is stolen by a masquerade attacker once he gains access either through legitimate users' ID and password or by using a keylogger. The alternative mechanism to such an attack is done internally due to the indolence of the user and or faith in the co-worker or friend where the user keeping the system open leave behind, the place either intentionally or by neglect or forgetfulness. In such a situation, the friend or colleague acts as a masquerade attacker.

- **Misfeasor**

Sameh *et al.* (2008), viewed that, Intrusion Detection Systems (IDS) in network security, also can be eluded by Misfeasors who are also recognized as insiders. The Misfeasors, as they are the insiders of an organization even if, have legitimate access to various data, programs, and or various resources to which they have a sound knowledge practically are not authorized to access such resources and they eventually misuse the same.

- **Clandestine User**

It constitutes a group of individuals who capture the supervisory control of the system and operate either below the level at which audit trail data is taken or can use privileges or system primitives to evade audit trail data being recorded. They may be either outsiders or insiders. It is difficult to trace out the identity of such users unless he activates his surreptitious operations either as a masquerader or as a misfeasor. Stallings (2011) has listed the following intrusion.

- Performing a remote root compromise of an e-mail server;
- Guessing and cracking passwords;
- Blemishing a Web server;
- Traffic regulation policy traces across the network, such as an unusually high rate of TCP connections;
- Perpetual Eco Policy;
  (Perpetual echoes on local port 7 and remote port 7. Mention may be made that, UDP port 7 is the echo port. In an attack, if the header specifies the source and target ports as port 7, the UDP datagram echoes back and forth between the local port 7 and the remote UDP port 7. When a perpetual echo occurs on port 7, IDS sends an intrusion notification to the Intrusion detection events page and the audit journal, but it does not send an e-mail notification).

- Viewing sensitive data including payroll records, medical information without authorization.
- Running packet sniffer on a workstation to capture username and password.
- Using a permission error on an anonymous FTP server to distribute pirated software and music files.
- Dialling into an unsecured modem and gaining internal network access.
- Posing as an executive, calling the help desk, resetting the executive's e-mail password, and learning the new password.
- Using an unattended, logged-in workstation without permission.

From the above discussions, it could be inferred that Network-based Intrusion Detection Systems (NIDS) monitor various activities on network segments as they sniff traffic when it flows over the network and alerts a security administrator in case of suspicion occurs. (Simpson, 2003).

The primary functions carried out by the IDS can be described as follows (Chen *et al.,* 2005).

- Monitor and Analyse user and system activities;
- Assess the integrity of critical system and data files;
- Recognise activity patterns reflecting known attacks;
- Respond automatically to detected activities; and
- Report the outcome of the detection process.

However, Kazienko and Dorosz (2003) while discussing the attack engulfed in Intrusion Detection System also has categorically spelled out the following security devices which are not included under the Intrusion Detection System.

- Networking logging system used to detect vulnerability to any DoS attack across a congested network as these are network traffic monitoring systems.
- Vulnerability assessment tools like CyberCop Scanner that is employed for bugs and flaws in operating systems and network services.
- Anti-virus products for detecting malicious software.
- Firewalls.

- Security/ cryptographic systems such as VPN, SSL, S/MIME, Kerberos, Radius, etc.

Further, the Intrusion Detection Working Group (IDWG), a devoted, skilled, and committed group defined a Common Intrusion Detection Framework (CIDF) based on four functional modules as discussed below. Figure-2.2 placed below elaborately visualizes the function of different modules.

- **E-Modules (Event Module)**
  It is a combination of various sensor elements that monitor a specific system so for analysis of the acquired information in other modules.

- **D-Modules (Database Module)**
  Here, the collected data from event modules are stored for further processing in Analysis and Response Modules.

- **A-Modules (Analysis Module)**
  This module processes the data analyzes the events and detects the potential hostile behaviour, and in the process generate an alarm in case the situation warrants.

- **R-Modules (Response Module)**
  This is the execution platform if any intrusion occurs of a response to perplexing the detected threat (Jyotsna and Ram Prasad 2011).

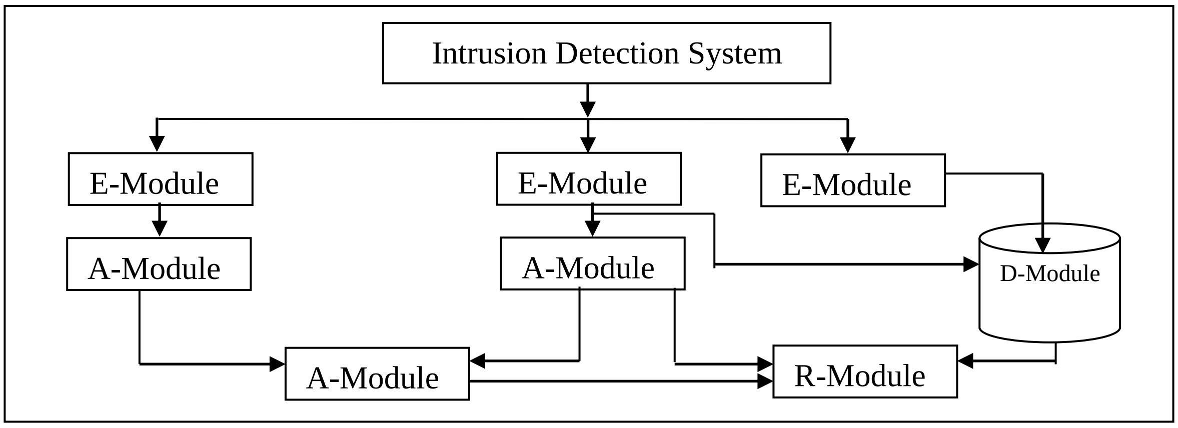


Fig -2.2: Common Intrusion Detection Framework

2.6.1 Need for Intrusion Detection System

Access to Internet has become pragmatic because of using multidimensional information for various purposes, materializing multifarious activities, etc. and at the same time, penetration of increasing threats, viruses, attacks, etc. cannot be ruled out in the systems. In this perspective protection of the systems from both outside and inside threats, attacks have become crucial and hence, the need for the Intrusion Detection System is well justified which provides reliance, security to the systems on information systems in a network environment. The adaptability of Intrusion Detection Systems has gained momentum for the security of the system in any organizational setup. The compelling reasons for gaining and use of Intrusion Detection System as pointed out by Bace and Mell (2001) have been discussed below.

i. Prevention of the systems from increasing the perceived risk of discovery and punishment for those who trail a destructive approach by attacking or otherwise abusing the systems;
ii. Observing attacks and infringements which are generally excluded from the purview of security measures;
iii. Detecting and handling attacks at the beginning which is invariably experienced as network probes and other extreme doorknob activities;
iv. Documentation of the existing threat to the organization;
v. Maintaining quality in security design;
vi. Developing improved diagnosis to detect intrusions.

2.6.2 Taxonomy of Attacks and Intrusions

The taxonomy of the Intrusion Detection System has been explained by various computer scientists in a multifarious way such as various attacks and intrusions which, however, does not have a consensus. Broadly, the Intrusion Detection Systems deal with hacking breaches leading to the performance of dangerous activities. Various terms as (i) Intrusion, (ii) Incident, (iii) Attack, signify Intrusion Detection System taxonomy. (Allen, 1999; Axelsson, 2000; Debar *et al.*, 1999; Jones and Sielken, 1999).

2.6.2.1 Analysis Strategy

It reflects the characteristics of the detector i.e intrusion detection engine. Lazarevic *et al.*, (2003) viewed that, the analysis strategy is called misuse detection when the Intrusion Detection System looks for events or sets of events that match a predefined pattern of a known attack. They also opined that the analysis strategy signifies anomaly detection when the Intrusion Detection System finds the intrusions as the unusual behaviour of the monitoring system.

2.6.2.2 Timing

Timing of IDS according to Base and Mell (2001) refers to the elapsed time between the events that are monitored and the analysis of those events and this can be categorized into two components.

- **Interval-Based (Batch Mode)**

Generally, the information in the interval-based Intrusion Detection Systems moves from the monitoring points to analysis engines in a continuous way leading thereby, the handling of information similar to store and forward communication schemes. Earlier, this pattern was being used in host-based IDS as they relied on operating system audit trails which were being generated as files. Interval-based Intrusion Detection Systems, however, prevent performing active responses.

- **Real-Time (Continuous)**

Real-time Intrusion Detection Systems being the paramount timing scheme for the network-based IDSs function incessantly to feed the information from information sources accumulated from network traffic streams. Here, real-time signifies processing control situations which otherwise mean that the real-time IDSs facilitate yielding instant results to permit the Intrusion Detection System to take action which affects the progress of the detected work.

2.7 Intrusion Detection System Approaches

Many approaches are involved in Intrusion Detection System. Some of them can be carried out through induction of the relevant software in the system while others can be implemented through strategies in the organization to minimize and prevent intrusion. Easttom (2011) has pointed out the following Intrusion Detection System approaches.

2.7.1 Preemptive Blocking

Preemptive relates to designing or having the power to deter or prevent an anticipated situation or occurrence. This approach is also known as banishment vigilance which is performed by way of noting down the trace of any dreadful activities in anticipation and blocking the IP address of the source or any users' system originating such an impending approach. In the event of tracing such an approach, the IP address of such systems requires blocking at the firewall to prevent intrusion. Normally, the software used to alert the administrator of such suspicious activity, and later on, the administrator after conforming to such approaches blocks the IP address. The software also blocks automatically such suspicious addresses. It should be noted that nothing prevents offending users from moving to a different system to continue their attack. Hence, such approaches form only a part of an overall intrusion detection strategy and not the entire strategy.

2.7.2 Infiltration

Infiltration in a system reckons to unauthorized entering to program code with evil intention to perform unsought and even concealed activities. According to AEC Data Security Institute, 80,000 types of infiltration are prevailing with a growth of 500 to 800 new types of infiltration appearing every month resulting thereby, constraints in deducing solutions to such emerging problems in the network along with differentiating from mutations of the types. However, based on the behaviour and program code constructions, the following types of infiltrations could be ascertained. It may be discussed that AEC is comparatively a highly risky Trojan horse that performs a series of harmful actions to a system, especially in a network environment. It seems to be undamaging but acts once the concerned application is installed on the users' computer. The trojan horse along with the inbuilt program is activated causing a series of damages. It has got two parts such as client part and the server part where the client part performs multifarious tasks on the infected systems, the server part transmits the data of the users and thus, the hacker is successful in his mission.

- **Trojan horse**

Trojan horse as discussed is application software that seems to be benign and secretly downloads the virus or some other type of malware to the system when connected to the network. Normally, it is a virus and flows to the systems through e-mails of the users. In the event of opening such malicious files, the system is damaged by way of,

i. Deleting data/files,
ii. Installation of a keylogger or other spyware to the system,
iii. Opening a backdoor for a hacker to use the users' information,
iv. Eavesdropping (Internet activity monitoring, password monitoring),
v. Disc formatting, Scoring out data, Effacing a random hard drive sector,
vi. Remote Access Trojan (RAT) – providing remote access,
vii. Furnishing remote access to sensitive data such as accessing password, code key, PIN, etc.),
viii. BOT program reacting to commands from the control server (robot) and
ix. Enabling DDoS attacks.

- **Worms**

These constitute independent programs or a set of programs without host code. These types of infiltration occur to the system through active worms that replicate and increase its functional copies on the Internet especially through e-mail, IRC, etc. BOT I-worm, a new worm employs social engineering and weak points of mail clients and infects the system by altering the .exe file to .pif. .scr., .ink, or doubling the affix to .txt .vbs, .jpg .exe, .zip .ex and changing the .exe file icon to WinZip etc. Further, it launches spontaneously on the operating system using its SMTP routine for spreading, extracting addresses of the victim's systems on the attacked hard drive in the windows address book, personal address book, temporary internet files, ICQ database, and other files.

- **Virus**

The virus, the dependent program code is connected to a host executable unit and launching of such executable units execute virus code which infects another executable unit by inserting its replication and spread to another system. A virus in the true sense of the term conks out a system as it instantly uses all available memory. Further, it flows in its capacity across the network. (Barker *et al*., 2001).

Mention may be made that, there are multiple types of viruses that cause infection to the system and many of them have slight variations of others. Some of the common viruses detected have been listed in the following Table- 2.2.

Table-2.2: Description of Common Virus

Sl.No.	**Types of virus**	**Description**
1	Boot Sector	Boot sector virus propagates once the system is booted. Such viruses are injected into the hard disk of the system through the infected pen drive, hard drives. While booting from an infected hard drive, the virus attempts to replicate itself onto any unprotected pen drives.
2	File Infected	Such viruses attach themselves to executable programs in the system. Further, it copies to the memory and attaches itself to any other executable files in the system once the infected executable is executed.
3	Polymorphic	It modifies them every time they move between the systems and it becomes difficult to detect.
4	Stealth	It hides them to prevent detection.
5	Encrypted	It encrypts to avoid detection.
6	Worms	It is a program and not a virus in the true sense, but it travels between machines and across the network connection.
7	Time bombs	It is a virus that executes a malicious act at a specific time.
8	Logic bombs	It executes a malicious act upon execution of a specific logical condition.

Source: Blacharski (1998). Network Security in a Mixed Environment, Foster City, CA, IDG Books.

An account of such other dreadful viruses of recent origin causing damage/defunct to the systems in one way or other is explicated in *Appendix-III.*

- **Hoax**

The hoax is defined as anything unreal. It is received through e-mails. Fake alarms e-mails use social engineering to send messages to the available addresses on the Internet. It is a type of infiltration that confuses the users and decreases their concertation causing them to neglect the real alarm messages of harm to the users' system.

Likewise, other infiltrations are caused through spams that are sent through infiltrated systems connected to the Internet (BOT) with a simulated heading making it difficult to find the senders' address and block the respective SMTP communications. Phishing is also a type of infiltration received through fraudulent e-mail on social engineering platform. It prompts the users to redirect the Uniform Resource Locator (URL) link, keylogger, etc. Further, it convinces the users to provide the username and password, personal details, bank details like credit card number, debit card number; pin, etc. pharming equally is a type of infiltration and similar to attack which insists the users redirect to the fake Internet banking sites by compromising DNS.

2.7.3 Intrusion Deflection

Over the network platform, intrusion deflection is gaining momentum in the security-conscious administrators. Easttom (2011) viewed that, intrusion deflection leads an intruder to believe that he has succeeded in gaining access to the system resources which rather he is tracked to a specially designed environment for observation and minimizing the harm (Halme and Baue 2012). This is usually recognized as a honeypot where the intruder accesses a fake system pretending to be the right information from the server created by the user.

2.7.4 Intrusion Deterrence

Deterrence refers to an act or process of discouraging actions or preventing occurrences by instilling fear or doubt or anxiety. From an intrusion perspective, it is to persuade an attacker to resist and fight to a standoff an ongoing attack. This is realized by increasing the perceived risk of negative consequences for the attacker. Two types of deterrence prevail as (i) Internal Deterrence and (ii) External Deterrence. Internal deterrence can be carried out in the form of login banners warning internal and external attackers of dreadful upshots while external deterrence could be affected by the laws against the computer/cybercrime (Axelsson and Sands 2006). Stallings (2011) in addition to the intrusion processes also has identified the following approaches.

2.7.5 Statistical Anomaly Detection

The system employs the data compiled from the previous network behaviour of licit users over a period. Statistical tests can be employed to the actual usage pattern to determine the level of confidence whether the behaviour is illegitimate. According to Farshchi (2003), anomalous activities are measured by several variables sampled over time and stored in a profile. The reporting process alerts depending upon the quantum of anomaly score of a packet. The reporting process alerts the user that the anomaly score of the packet is greater than or equal to the threshold value level set by the user. (Manikopoulos and Papavassiliou, 2002). This anomaly detection is further grouped into two types of approaches such as,

- **Threshold Detection**

Farshchi (2003) viewed that, the anomaly score is allocated after evaluation of the source IP, source port, destination IP, and destination port, among others. The spade depending upon the user-specified threshold level either flags the packet or allows it to pass through the network without notification. If the setting of the threshold in Spade is too high, the user will miss critical packets and if it is too low, then the analyst will see many false-positives. Spade, however, has an alternative that performs automatic threshold adjustment to let Spade decide the critical threshold number. It also generates other reports of importance such as a survey about the distribution of anomaly scores and the feature statistics such as entropy and conditional probabilities.

- **Profile-Based Detection**

It is concerned with developing the activity of each user for using to detect changes in the behaviour of individual accounts. The intrusion detection in such techniques is performed by observing events in the computers/systems through a set of rules which ultimately lead to deciding suspicious in a given patter activity. However, even if there is some overlapping, all approaches can be characterized either as anomaly detection or as penetration identification even if, having some overlapping (Stallings and Brown, 2007).

2.7.6 Rule-Based Anomaly Detection

Rule-based techniques perform the job of intrusion detection in the system by observing events and employ a set of rules that lead to a decision regarding suspiciousness of a given pattern or behaviour. It is further divided into two types as follows.

- **Anomaly Detection**

The rules are framed to detect any deviation/alternation/modification from previous usage patterns. The rule-based approach facilitates analyzing the historical audit record to find out the usage patterns. The rules represent the past behaviour pattern of the users, programs, privileges, time slots, terminals, etc. It is alike in the terms of its approach and strengths to statistical anomaly detection. The knowledge of security vulnerabilities within the system is not required in the case of rule-based anomaly detection (Bhati and Rai, 2016).

- **Penetration Identification**

It is related to an expert system technology to search for suspicious behaviour. Such systems use rules for identifying the known penetrations or penetrations that would take advantage. Here, the experts use their expertise to bring forth rules rather than through automatic analysis of audit records. Thus, to sum up, the statistical approaches seek to define normal or expected behaviour while rule-based approaches try to define the proper behaviour. Further, statistical anomaly detection is effective for masqueraders who are unlikely to mimic the behaviour patterns of the accounts they appropriate while, rule-based approaches recognize misfeasors. But in practice, a system applies both approaches against a range of attacks.

2.8 Audit Records

Audit records are an effective tool in intrusion detection where on-going activities of the users are maintained as input to an Intrusion Detection System and it is performed basically in two plans such as (Stallings, 2011),

- **Native Audit Records**

The accounting software used in the multiuser operation system gathers information on user activity and maintains a record which is known as Native Audit Records. While using such information, no additional software is required but the information in such records may not be in a convenient form.

- **Detection-Specific Audit Records**

It is related to a collection facility that can be enforced for generating audit records that constitute information as required by the Intrusion Detection System. Regardless of its type, a record constitutes the fields like (i) Subject, (ii) Action, (iii) Object, (iv) Exception Condition, (v) Resource Usage, and (vi) Time Stamp (Kahate, 2011).

2.9 Architecture of Intrusion Detection System

Intrusion detection is endured with numerous architecture abstractions that reveal strengths and weaknesses with regards to various factors like efficiency, security, integrity, durability, and cost-effectiveness (Roque,users.cis.fiu.edu/). The architecture of IDSs is used to differentiate between centralized IDSs that analyze the data collected only from a single monitored system and distributed IDSs that collect information from multiple monitored systems to investigate global, distributed, and coordinated attacks. The basic architecture of IDS has been discussed in the following Figure-2.3.

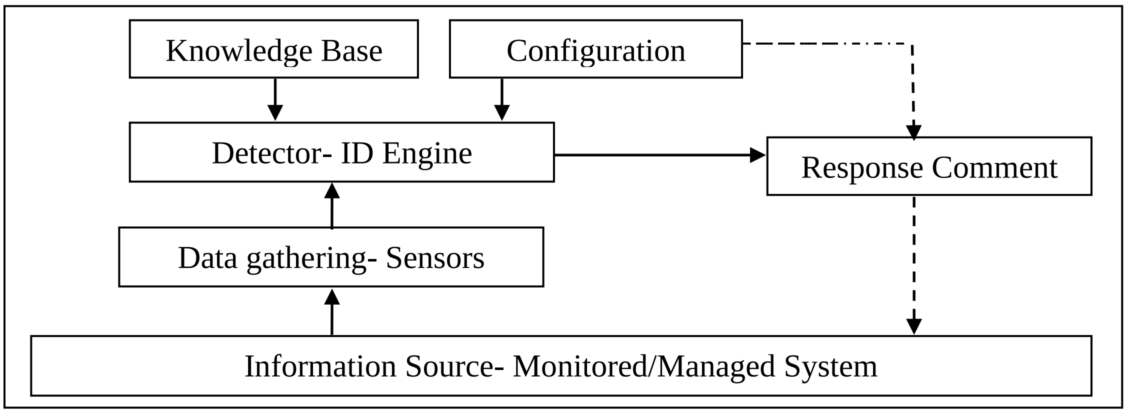


Fig.- 2.3: Basic Architecture of Intrusion Detection System

2.9.1 Types of Intrusion Detection System

The Intrusion Detection System can be divided into several types based on the type of systems being monitored (Biermann *et al.*, 2001). Intrusion Detection concentrates on observing incompatible, erroneous, and anomalous activities in the network-based system. Whitman and Mattord (2005) viewed that, Intrusion Detection System connotes not only to a process of monitoring the events that occur in a computer system or network environment but also analyze them for signs of possible incidents, which signify violations or imminent threats of violation of computer security policies, acceptable use policies or standard security practices. An Intrusion Detection System is a device or software application employed for monitoring a network and/or information system for detecting malicious activities or policy violations and responds to that suspicious activity by warning the system administrator in many ways, including displaying an alert, logging the event or even paging the administrator. (Patel *et al.*, 2010). The IDS which implement the detection of such extraneous, redundant intrusion in the system in a network-based platform can be broadly be classified in three ways as discussed below.

2.9.1.1 Host-Based Intrusion Detection System (HIDS)

Host-based Intrusion Detection System reckons upon a single host or computer system and the events occurring within that host for suspicious activity (Lichodzijewski *et al.*, 2002). It is implemented by placing a sensor on a particular computer system. The Host-based IDS also monitors wired and wireless network traffic on the host including its network activities, system logs, running processes, file access, and modifications including system and application configuration changes. The Host-based Intrusion Detection System includes the major components of the technologies including the architectures used typically for deploying various components. This also examines the security capabilities of the technologies critically along with the methodologies used to identify suspicious activity. The Host-based use audit trails as the source of choice for intrusion detection information as it protects the operating systems in its audit layer and provides detailed information (Graff, 2001 and Gautam, 2016).

Securities are the prime concern for the system, various components are employed in the Host-based Intrusion Detection System and they primarily include, (i) Sensors/ Monitors; (ii) Agent; (iii) Management Server; (iv) Database Server; and (v) Console. However, (Wagner *et al.,* 2002) discussed that most of the Host-based Intrusion Detection Systems have detection software that is known as agents. Each agent is primarily concerned with monitoring various activities on a single host and performs as a prevention of the systems where Intrusion Detection Systems are enabled. He further viewed that, some of the Host-based Intrusion Detection Systems products instead of installing agent software on individual hosts apply dedicated appliances running agent software. The appliances are so positioned that they can monitor effectively the network traffic in the duplex communication of a host. These appliances, however, are more technically concerned with the Network-based Intrusion Detection System due to their deployment to monitor network traffic. The central management server over the network controls the agent software including monitoring the agent configuration and collects various events from the agent software. The central Host-based Intrusion Detection System server after collecting events correlates the activities from all its monitored hosts based on predefined signatures and customized rules to produce alerts on suspicious or malicious behaviours. Further, the collected events are sent to log correlation software such as the ISP Log Correlation program for a critical analysis. Moreover, the agent so employed on the network traffic significantly protects either a server or a client host, or an application service.

A Host-based Intrusion Detection System provides enormous information which includes user authentication, file modifications and or deletions, etc. Hence, it is designated as secondary protection to devices on the network. Various products of a Host-based comprise Tripwire (Kim and Spafford, 1995), SWATCH (Bauer, 2003), Event Monitoring Enabling Responses to Anomalous Live Disturbances *(*EMERALD), etc. (Dolezelova *et al*., 2017).

2.9.1.1.1 Network Architecture of Host-based Intrusion Detection System

Network architecture is a framework in which the Host-based Intrusion Detection System works. The agents on the network of the organization are distributed in the host to communicate over the same network. Most products encrypt their communications preventing eavesdroppers from accessing sensitive information.

2.9.1.1.2 Host-Based Intrusion Detection System Agents

The agents in the Host-based Intrusion Detection System are deployed to critical hosts such as publicly accessible servers which constitute sensitive information (Bivens *et al.*, 2004). The agents are available in servers, systems and positioned effectively to perform effective communication. Sen (2010) viewed that, the system monitoring agents are responsible for collecting, transforming, and distributing intrusion-specific data on request and elicit information collection procedures and the agents publish the details of the variables they monitor, which can be utilized by other agents. Further, two categories of communications among Agents are available i.e. (i) Communications among agents residing at the same host, and (ii) Communication among agents on different hosts. However, the agents installed in the Host-based Intrusion Detection System on endpoints can visualize the unencrypted activity while, in a Network-based Intrusion Detection System, the agents cannot analyze the activity within encrypted network communication. The agents as reflected in the following Figure-2.4 depicts their deployment in Host-based Intrusion Detection System architecture.

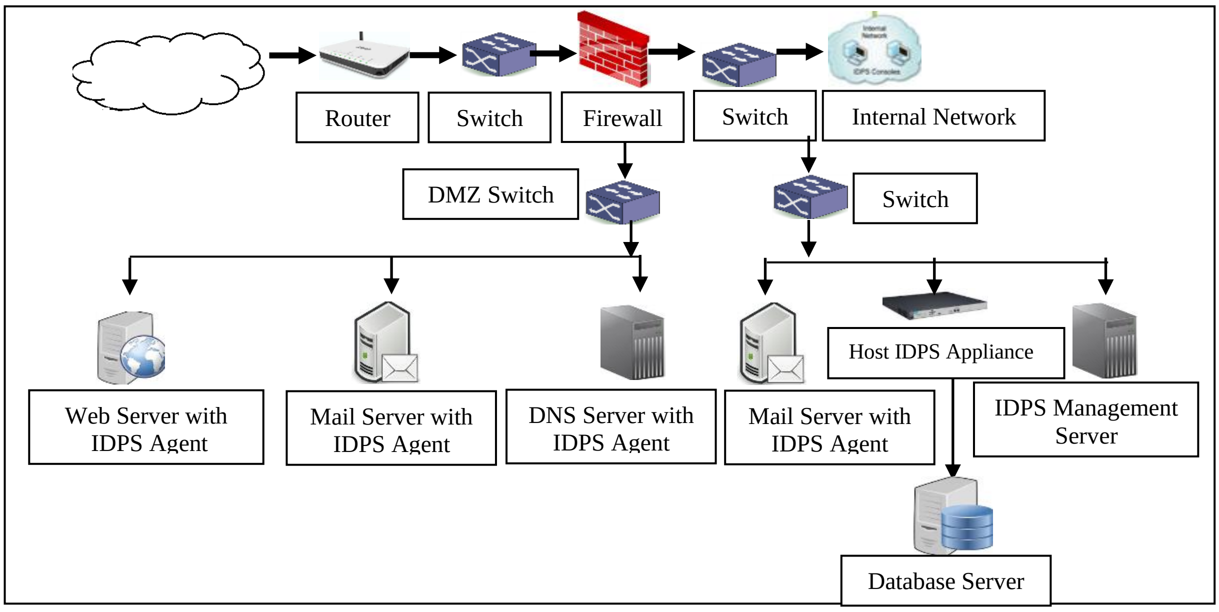


Fig- 2.4: Host-based Intrusion Detection System Agent Deployment Architecture

2.9.1.1.3 Host Architecture

The Intrusion Detection System agents installed in the hosts modify the architecture to facilitate intrusion prevention capabilities which are performed through a shim that constitutes a layer of code between the existing layers of codes. A shim intercepts the data at a point so that it can be moved conveniently from one piece of code to another and analyzes the data for permitting or prohibiting the same. A shim, in the Host-based Intrusion Detection System, is used for different types of resources along with network traffic, system calls, file system activity, windows registry activity, and other applications like e-mail, etc. (Scarfone and Mell, 2007).

However, some agents in the Host-based Intrusion Detection System without any change the host architecture and shim monitor or analyze the artifacts the activity. The Host-based Intrusion Detection System solutions can be performed either through installing agents on hosts or the use of agent-based contraptions where installing agents on hosts is preferable for detection and prevention due to direct access of the agents to the host's characteristics. Further, the agents support to few operating systems and in the event of failure of support to an operating system by an agent, the appliance can be positioned. Another reason for the use of an appliance is due to a negative impact on the performance of a monitored host by an agent (Scarfone and Mell, 2007).

2.9.1.1.4 Security Capabilities in Host-based Intrusion Detection System

Security has become fundamental especially in an online environment to protect the system, database, information, etc. Intrusion Detection and Prevention System (IDPS) happens to be a viable podium which principally concerns with security by way of detecting possible incidents, logging information, and finally transmit the information to the security administrator. The other purposes of using IDPs are to detect problems associated with security policies, recording the threats, and discouraging the personals while offending the security policies. It may be discussed that, while, Intrusion Detection System, a software that automates the Intrusion Detection Process, Intrusion Prevention System which is also the software is associated with the capabilities of Intrusion Detection System and preventive measures.

2.9.1.1.5 Advantages and Disadvantages of Host-based Intrusion Detection System

- **Advantages**

Multiple advantages as discussed below are included in the Host-based Intrusion Detection System even if it not so robust likes the Network-based Intrusion Detection System. The Host-based Intrusion Detection System,

i. Provides exhaustive information during an attack in a system;

ii. Monitors events local to a host leading thereby, detecting attacks that are invisible by a Network-based Intrusion Detection System.

iii. Operates in an encrypted network traffic environment.

iv. Detects and also prevents attacks that involve software integrity breaches like, Trojan Horses.

v. Allows fewer false alerts than produced by Network-based Intrusion Detection System.

vi. Does not have a consequential effect on the switched network.

vii. Monitors local files for any changes or modifications and or deletion.

viii. Visualizes the outcome of an attempt at a targeted attack by an attacker who directly accesses and monitors the data files including an operating system (Lichodzijewski *et al.*, 2002).

Over and above, the Host-based Intrusion Detection System provides the following information while detecting intrusion (Cichonski *et al.*, 2012)

i Date and Time;

ii Sensor IP Address;

iii Source and destination of IP Address;

iv Source and destination of port numbers;

v Name of specific attack;

vi Description of the attack type;

vii Network protocol used;

viii Attack severity level;

ix Type of loss expected;

x Type of vulnerability exploited;

xi Input validation (Buffer overflow or Boundary condition);

xii Access validation (Faulty Access Control Mechanism);

xiii Exceptional condition;

xiv Environmental (unexpected interaction with an application and the operating system or between two applications);

xv Host configuration;

xvi Race (Delay between the time of checking by a system and time it operates);

xvii Design;

xviii Types of software and versions vulnerable;

xix References to advisories about the attack or vulnerability etc.

- **Disadvantages**

The disadvantages of the Host-based Intrusion Detection System are discussed below (Thomas, 2009).

i Consequent upon the prevailing of network heterogeneity and multiple operating systems, a single Host-based Intrusion Detection System do away with translating all operating systems including network applications and file systems.

ii In the absence of a corporate key, a Host-based Intrusion Detection System cannot decode encrypted information.

iii Host-based Intrusion Detection System relies upon the system created audit record which is exhaustive by nature both in terms of quality and quantity leading thereby the effectiveness of Intrusion Detection System.

iv Due to proliferated networks, thousands of workstations operate in the network which makes it difficult for a Host-based Intrusion Detection System to install in each system for monitoring leading thereby both expensive and unmanageable.

v Host-based Intrusion Detection System detects the intruder only when the intruder bypassing the security measures reach the monitored host system.

2.9.1.2 Network-Based Intrusion Detection System (NIDS)

Network-based Detection also known as Network-based IDS (NIDS) examines each node on the network under observation. It is primarily associated with monitoring network traffic to determine the network segments or devices for suspicious activities by analyzing the network especially in local area networks, transport, and the application protocols (Sommer, 2008). Invariably, a network-based intrusion detection system places its reference monitor in the kernel/user layer and watches for anomalies in the system which is known as patterns. The advantages of Network-based intrusion include (Chen *et al.*, 2005), (i) No processing impact on the monitored hosts, (ii) The ability to observe network-level events, and (iii) Monitor the entire segment at once.

The Network-based Intrusion Detection Systems (NIDS) collect input data by monitoring network traffic such as packets captured by network interfaces in promiscuous mode whereas, the Host-based Intrusion Detection Systems (HIDSs) rely on events collected on the hosts they monitor. Further, the Network-based

Intrusion Detection System examines activity on the network and has visibility into the traffic crossing the network link to which it monitors but excludes the happening in another individual system. The Host-based Intrusion Detection System is concerned about examining the activity in the individual system like web server, mail server but it is not concerned about the outer domain than the individual system. (Bae *et al.*, 2012). In both the Intrusion Detection Systems, several components work together which has been illustrated in Figure-2.5.

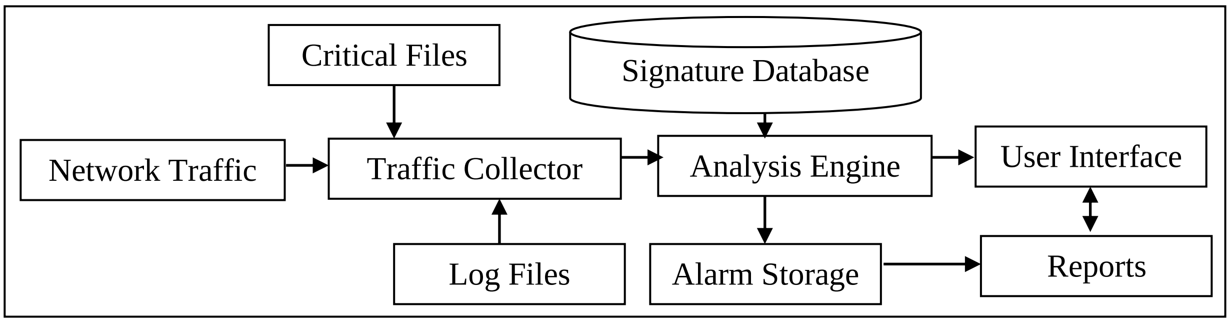


Fig.- 2.5: Logical Depiction of Intrusion Detection System components
Source: http://www.ucvts.tec.nj.us/cms/lib5/NJ03001805/Centricity/ Domain/241 /ch13.pdf.

Mention may be made that, the signature database depicted in the above figure relates to a collection of patterns and definitions of known suspicious or malicious activity and analysis engine, the brain of the Intrusion Detection System examines the collected network traffic and compares it to the known patterns of suspicious or malicious activity stored in the signature databases.

An Intrusion Detection System constitutes three logical components (Stallings and Brown, 2010).

- **Sensor (Traffic Collector)**

The sensor is also known as a Traffic collector is a hardware component that provides the system with information about its location, surroundings, etc. It is responsible to collect data and contains evidence of an intrusion. (Ahdi *et al.*, 2012). The sensors act as agents that facilitate monitoring hosts or networks on a real-time basis. It can be mentioned that the database of attack signatures in the Intrusion Detection System are the patterns of different types of previously detected attacks. The role of the sensor is that, once it detects the malicious activity matches the malicious packet against the attack signature database and when it confirms the availability of such malicious packets in the database then apart from reporting about such packets to the management console also takes different actions based on their configuration. In a Host-based Intrusion Detection System i.e, HIDS, the sensor acts as log files, audit logs, or traffic coming to or leaving a specific system while in a Network-based i.e, NIDS, it is related to copying traffic off the network link and function as a sniffer.

- **Analysers**

  The analyzer in an Intrusion Detection System takes the responsibility for performing intrusion analysis of information that may be representative of vulnerabilities in and misuse of IT resources (National Security Agency, 2001). The analyzer in the process after receiving inputs from more than one sensor confirms the conclusion about the occurrence of the intrusion with evidence. Further, the analyzer provides adequate guidance for taking actions of such intrusion (Stallings and Brown, 2010).

- **User Interfaces**

  The user interfaces in an Intrusion Detection System enable a user to view the result from the system or control the behaviour of the system. Exerting control over the network administrator from a common outline of the complete network behaviour, the user interface access lower level (more detailed) data in a structured database starting from a higher level to an amassed view of NIDS data and exhaustive analysis of the reasons of the anomalies, alarms, and elusive network elements (Zahariev (2011).

2.9.1.2.1 Architecture and Components of Network-Based Intrusion Detection System (NIDS)

Architectural design and induction of various components are crucial in the Network-Based Intrusion Detection System as components have an interrelated function. The components broadly comprise sensors, agents, database servers, management servers, and consoles where the sensors and agents both monitor and analyze the activity. Primarily, sensors and agents are used to monitor the network and hosts respectively. Further, while, the management servers perform as receivers of information from sensors, the database servers act as repositories of event information recorded by the sensors, agents, and management servers. Consoles, which are the programs, facilitate interfaces for Intrusion Detection and Prevention System for both the users and the administrators (Scarfone and Mell, 2007). These components have relative functions and are designed in the network.

The Network Interface Card (NIC) is equally a prime component in the Network-Based Intrusion Detection System which is deployed into a promiscuous mode for monitoring and accepting the incoming packets. Hence, unique sensors are deployed in large numbers in a network for Intrusion Detection and Prevention Systems. Sensors are categorized in two formats (Scarfone and Mell, 2007) such as (i) Appliance-based sensor comprising of hardware specially designed and the sensor software and (ii) Software only.

The architecture of the Network Intrusion Detection System comprises five components as discussed below that function in an integrated way in the network. In such a scenario and for better performance of the results, all the components are coupled together and developed in a single software with a logical sequence but separated in their function.

- **Collector**

  The collector facilitates an interface for accessing data used in the detection process. Network Tap is a kind of data collector in the Network Intrusion Detection System (NIDS) which renders access to all raw network packets.

- **Detector**

  The detector, a potential component is a brain that conducts the actual detection process in NIDS. The crucial function of the detector is that it decides the future course of action after accessing the data from the collector.

- **User Interface**

User Interface pertains to the delivery of device policy of Network Intrusion Detection both through ASCII files and Graphical Users' Interfaces.

- **Storage**

Data invariably is either obtained by the detector or through external means. Further, storage leverages a database system (Sommer, 2008) and thus, data acquired from storage are useful for forensic analysis and it conforms to the successful intrusion.

- **Responder**

The responder resists detected intrusion for future injury. A response is actuated both automatically and or manually through the user interface. Invariably the active responses do not allow the connectivity to the potential attacker and also to the counter-attackers. The communication links of the general architecture of NIDS are shown below in Figure-2.6.

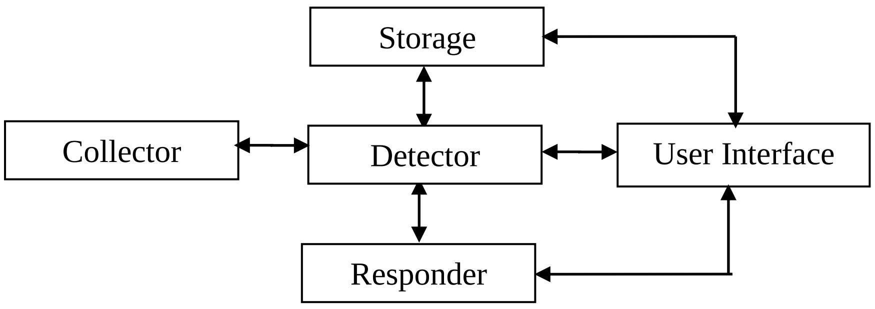


Fig-2.6: Communication Links of General Architecture of NIDS

While deploying the different components in a network environment, the collector needs installation at the upstream link of the network to facilitate the detector to examine the call traffic which, however, cannot detect the packets exchanged between the internal hosts. The sensor practically performs a combined collector/detector in this environment. Further, multiple installations of collectors are required with detectors for various performances in a single operating environment and thus in the process, several schemes use multiple sensors and one central detector which is broadly identified as Sensor Model. The typical positioning of various components in a Network Intrusion Detection System is explained in Figure-2.7.

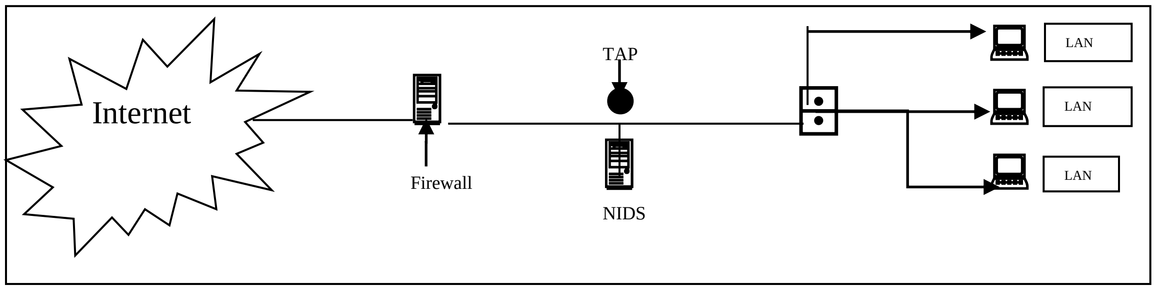


Fig-2.7: Typical Deployment of a Network Intrusion Detection System

2.9.1.2.2 Security Capabilities

The products of Network-based Intrusion Detection and Prevention System render wide dimensions of security capabilities which, however, can be identified as (i) Information Gathering Capabilities, (ii) Logging Capabilities, (iii) Detection Capabilities and (iv) Prevention Capabilities, etc. Network-Based Intrusion Detection and Prevention System also stresses providing Security Information and Event Management (SIEM), a software designed to import information from various security-related logs and correlate events among them (Kent and Souppaya, 2006).

- **Information Gathering Capabilities**

Information gathering capabilities are essential for the organizations to identify their Intrusion Detection and Prevention Systems detection methodologies and analysis function and evaluate each IDPS product under consideration for its ability to offer those capabilities (National Computer Board, 2011). Some network-based IDPSs proffer restricted information-gathering capabilities. This means that they can collect information on hosts and the network activity involving those hosts. (Scarfone and Mell, 2007).

- **Logging Capabilities**

Network-Based Intrusion Detection and Prevention System execute logging of data associated with observed events where data could be used to affirm the cogency of alerts, look into incidents, and correlate events between the Intrusion Detection and Prevention System along with other logging sources (other security controls, OS logs). Intrusion Detection and Prevention System log elementary information at a minimum, like a date and time, event or alert type, source of the event, the sensor or agent that detects the event. Every Intrusion Detection and Prevention System log supportive data associating the details of the event and these data fields are particular to the product types of Intrusion Detection and Prevention System. Intrusion Detection and Prevention System also provides a mechanism to allow users to associate each log entry with corresponding external reference including Common Vulnerabilities and Exposures (CVE) numbers that provide universal identifiers for vulnerabilities and other references vendor security advisories (National Computer Board, 2011). (Scarfone and Mell 2007).

- **Detection Capabilities**

Network-based Intrusion Detection and Prevention Systems offer extensive and broad detection capabilities. The detection capabilities are important for many implementations. Most Network-Based Intrusion Detection and Prevention System products use a combination of Signature-Based Detection, Anomaly-Based Detection, and Stateful Protocol Analysis Techniques to perform an in-depth analysis of common protocols. Organizations use such Network-Based Intrusion Detection and Prevention System products that use a combination of such techniques. Usually, the detection methods are interlacing. Mention may be made that, a Stateful Protocol Analysis engine analyses activity into requests and responses, each of which is examined for anomalies, and the same is compared with signatures of known bad activity. Some products also use the same techniques and provide the same functionality as Network Behaviour Analysis (NBA) software (Scarfone and Mell, 2007).

- **Prevention Capabilities**

Invariably, the sensors deployed in a Network-based Intrusion Detection and Prevention System create hindrances to an intruder, and the sensor is applied in three ways i.e, a) Passive, b) Inline, and c) Both passive and inline. Phenomena associated with a passive sensor is that it ends the current TCP session by sending the TCP reset packets to both the endpoints and this process is also known as session sniping. In inline, the deployment of the sensor in Intrusion Detection and Prevention System obstructs the intrusion in three ways as, (i) Firewall which rejects or prevent the suspicious network activity, (ii) Restraining Bandwidth Usage for inappropriate use of protocol leading to DoS attack, malware distribution, or peer-to-peer file sharing and (iii) Altering Malicious Contents where, vicious i.e, the infectious contents are substituted with genuine contents and the sanitized packet are sent to its destination. Here, the sensor automatically normalizes all the traffics in a network environment by repackaging application payloads in new packets. In passive and inline, reconfiguring other network security devices such as routers, firewalls, switches, etc. are required to wedge the functioning of unusual activity of the devices (Kabila, 2008).

2.9.1.3 Vulnerability Assessment Intrusion Detection System (VAIDS)

A good Intrusion Detection System is sensitive as it has a large rule database as opposed to anomaly detection only and it can spot potential attacks (Philips and Swiler, 1998). It is essential to receive all types of alerts in a network environment and hence, Security Metrics Appliance is applied which integrates vulnerability assessment with its Intrusion Detection System. It can be explained that, when a system is being attacked, it spontaneously finds at the last vulnerability assessment database for the attack target, and then, the analysis is originated to determine if the target is vulnerable to attack. In the absence of vulnerability on the target, no alert is sent to the administrator which, however, sends an alert to the administrator in case of the presence of vulnerability on the target (Reddy, 2013).

In a network environment (Sivanandam *et al.*, 2006) mentioned the Vulnerability Assessment Intrusion Detection System has become pragmatic which detects vulnerabilities on internal networks and firewalls and these are two important models as discussed below to analyze events to detect attacks.

- **Misuse Detection Model**

In this model, the Intrusion Detection System detects intrusions by looking for an activity that corresponds to known signatures of intrusions or vulnerabilities.

Most of the Intrusion Detection System commercial tools refer to the Misuse Detection Model and signatures of Intrusion Detection System are always being updated by the vendors.

- **Anomaly Detection Model**

Here, the Intrusion Detection System detects intrusions by searching abnormal network traffic. The Intrusion Detection System based on Anomaly Detection Model can detect the symptoms of attacks without specifying the model of attacks, but they are sensitive to false alarms.

2.9.1.4 Distributed Intrusion Detection System (DIDS)

Intercommunication is established among various individuals or firms and it is established over a platform of a large network. The distributed Intrusion Detection System comprises numerous Intrusion Detection Systems in such a podium. These activities are performed through a central server that facilitates advanced network monitoring, incident analysis, and instant attack on data (Einwechter, 2001). The distributed Intrusion Detection System extends both the host-based and network-based type of defence towards intrusion and it consists of three following modules that are intended for preventing intrusion on a large scale (Snapp *et al.*,1991).

(i) Host agent module,

(ii) LAN monitor agent module,

(iii) Central agent module.

To explain briefly, while, the host agent module is associated with generating logs about a single host and sends them across the internet to a central manager module. A LAN monitor agent module generates logs regarding traffic on the local area network that it is monitoring. It also sends its logs to the central manager module. The central manager module gathers these data and data from other LANs and hosts. (Zimmermann *et al.*, 2006).

The DIDS system not only provides a feasible platform for quick instantaneous and adaptive techniques to recognize the harmonized attacks across numerous network segments but also traces the manners of the attackers. It also saves the deployed networks of the corporation in terms of money by reducing the number of incident analysts and the time as well required to collect logs from the various Intrusion Detection Systems. It stores the attack records in a single place and allows the analyst flexibility in discovering attack patterns, and other unnoticed attacks (Einwechter, 2001).

2.10 Intrusion Detection and Prevention System

Principally, the Intrusion Detection and Prevention System can be divided into four types such as (Scarfone and Mell, 2007).

(i) Network-based Intrusion Detection and Prevention System

It is associated with not only monitoring network traffic of a network or devices but also recognize suspicious activity by analyzing the network and application protocol activity.

(ii) Host-based Intrusion Detection and Prevention System

It is concerned with monitoring not only the characteristics of a single host but also the events occurring within the host for suspicious activity.

(iii) Wireless Intrusion Detection and Prevention System

It is related to the concept of monitoring wireless network traffic including analyzing for identification of suspicious activity in the networking protocols.

(iv) Network Behaviour Analysis (NBA)

It pertains to the examination of network traffic and recognizes threats that generate traffic flows such as DDoS attacks which are malware and policy violations.

While discussing the Host-based Intrusion Detection and Prevention System it can be mentioned that it provides a wide range of security capabilities that can be divided into the following four classes (Scarfone and Mell, 2007).

① **Logging Capabilities**

Usually, Host-based Intrusion Detection and Prevention System execute logging of data related to detected items which are applied for many purposes such as, (i) Confirmation of the validity of alerts, (ii) Investigate incidents, and (iii) Correlate events between Host-based Intrusion Detection and Prevention System and other logging sources. The Host-based Intrusion Detection and Prevention System log various data fields which include,

- Timestamp i.e. date and time;
- Event or alert type;
- Rating which includes, priority, severity, impact, confidence, etc.;
- Event details specific to the type of events like IP address and port information; application information, filenames and paths, and user IDs; and
- Performance of Prevention action.

② **Detection Capabilities**

The Host-based Intrusion Detection and Prevention System potentially detect malicious activity. A combination of both signature-based detection techniques and anomaly-based detection techniques is used for identifying known attacks and previously unknown attacks respectively. Further, the anomaly-based detection techniques also specify policies and or rules set for the purpose. The detection capabilities in Host-based Intrusion Detection and Prevention System comprises, (i) Types of Events Detected, (ii) Detection Accuracy, (iii) Tuning and Customization, and (iv) Technology Limitations.

2.11 Comparison of HIDS and NIDS

A comparative study of the various functions of HIDS and NIDS is depicted below in Table- 2.3, which visualizes a clear distinction between the two Intrusion Detection Systems (Magalhaes, 2003).

Table- 2.3: Comparison of HIDS and NIDS

Sl No.	Function	HIDS	NIDS
1	Protection on LAN	Protect Host computer	Protect LAN
2	Protection of LAN	Protects off the LAN	LAN protection only
3	Machine Registry Scans	Yes	No
4	Versatility	More versatile system	Less versatile system
5	Implementation	Easy to implement	Easy to implement
6	Training Requirement	Less	More
7	Cost of ownership	Less in the long run	High
8	The requirement of Bandwidth on LAN	No	Yes
9	Network Overload	No	Double the total network bandwidth required for LAN
10	Spanning Port Switching Requirement	No	LAN traffic is scanned
11	Update Frequency to clients	Update all clients from a central file	No
12	Cross-Platform Computability	Specific to Operating System, Application	Adaptable to cross-platform environments
13	Alarm Function	Alarm the individual or the Administrator	Alarm the individual or the Administrator
14	Packet Rejection	No	Reject or drop packets
15	Specialists Knowledge	Application of specific Knowledge	Knowledge is required for installing and understanding a network security
16	Central Management	Specific to the Host with less central management	Centrally managed

17	Disable Risk Factor	The failure rate is low	The failure rate is high as one point of failure

2.12 Machine Learning Approaches- Network Intrusion Detection

Broadly, there are two approaches to intrusion detection such as (i) Anomaly Detection and (ii) Misuse Detection. In anomaly detection, initially, the systems learn the normal activity profile and flag all the system events that do not match with the established profile. But all identified types of attacks i.e. intrusions in misuse detection can be discovered by looking into the predefined intrusion patterns in system audit traffic. While the vantage of the anomaly detection concentrates on the capability to recognize the novel (or unforeseen) attacks at the expense of a high false-positive rate, the misuse detection is its potentiality finds difficulty in detecting the new or unanticipated attacks for high detection rate (Panda *et al.,*2011).

In continuation to the above Intrusion Detection System, the most suitable and acknowledged type of approach-based intrusion detection also prevail based on various implications and behaviour in a different environment and it can apply in two ways as, (i) Misuse Detection and (ii) Anomaly Detection which has been discussed below (Sayar *et al.*, 2014).

2.12.1 Misuse Detection

Misuse detection is principally concerned with analyzing the accumulated information for the specific attack of the documented signature from a large database (Carr *et al.*, 2010). This is also identified as a signature-based and known attack which is equipped with a library of attack patterns each time when such a pattern is found in the network stream (Sommer, 2008). The attack patterns can be well depicted in various semantic levels. Specific byte sequences are a most admissible representation of such stream that frequently is available in raw payload stream of attack connection and these are prevalent mostly in Network Intrusion Detection. The misuse detection is also known as a Signature-based system. In IDS terminology, it downplays false positives. (Patel *et al.*, 2010) pointed out that, the misuse detection schemes signify attacks either in the form of a signature or a pattern to facilitate detecting variations of the same attack. This otherwise means that the systems are not unlike virus detection systems and can detect many or all known attack patterns, but they are of little use as yet attack methods are unknown. Further, misuse detection systems try to recognize known “bad” behaviour. The primary concern of the misuse detection is associated with (i) write a signature that embraces all possible variations of the relevant attack and (ii) write signatures that are incompatible with non-intrusive activity (Newman *et al.,* 2004). The advantage of misuse detection is that it is associated with focussing analysis on the audit data and produces few false positives while, the disadvantage is that it is lacking the ability to detect the invented attacks and hence, signature databases require updating and Intrusion Detection System must be able to compare and match activities against large collections of attack signatures (Patel *et al.*, 2010).

Further, it is observed that the signature-based Intrusion Detection Systems are more vulnerable to attacks that focus to actuate a good quantum of detection alerts. These are performed by injecting traffic that matches the signatures used in the analysis process. Such attacks can be performed not only by exhausting the resources on Intrusion Detection System computing platforms but also to hide the attacks available within many alerts produced. However, in contrast to firewalls, a misuse-based Intrusion Detection System scans all the packets at both 3rd and 4th layer along with the application-level protocols looking for backdoor Trojans, Denial of Service Attacks, Worms, Buffer Flow Attacks, etc. and detect scans against the network (Patel *et al.*, 2010). Thus, Intrusion Detection Systems facilitate wider options and visibility to detect signs of attacks and compromised hosts. Despite this, the need for a firewall to block traffic before encroachment the network is seriously been felt essential.

2.12.2 Anomaly Detection

An anomaly-based Intrusion Detection System is explicated as a deviation from the normal or expected behaviour. According to (Lockhart, 2007) another

Network IDS is a Statistical Monitor that also monitors the traffic on the network which, however, maintains a statistical history of the network packets rather than searching for a pattern and report when they come across a packet outside the Normal Network Traffic Pattern. Further, anomaly detection techniques rely on models of the normal behaviour of a system, and the models such as the applications or the network focus on the users. Anomaly Intrusion Detection is also known as Behaviour Based System which detects undesirable traffic.

Thomas (2009) viewed that the basic premise of anomaly intrusion detection systems is that attacks differ from usual behaviour both in type and amount. Further, the author discussed that the advantage of detecting previously unknown attacks is paid for in terms of high false-positive rates in anomaly detection systems and it is difficult to train an anomaly detection system in a dynamic environment. The anomaly detection systems involve essentially composite factors. While comparing the above two intrusion detection techniques it could be mentioned that, while misuse detection techniques are implemented as a first line of defense, the anomaly detection techniques can be used as a second line (Chen *et al.*, 2005).

A comparative study of Misuse Detection and Anomaly Detection placed below in Table-2.4 indicates the difference (Zanero, 2007) followed by a detailed description of Machine Learning Methods for Network Intrusion Detection in Table-2.5 below.

Table- 2.4: Comparative study of Misuse Detection and Anomaly Detection

Sl.No.	**Description**	**Misuse - Based Detection**	**Anomaly-Based Detection**
1	Update	Required continuously	Not required
2	Training	Not required	Extensive and Complex training required
3	Tuning and Alteration	Required	Tuning is incorporated into the training process
4	Identification of new or novel attacks	Unable to identify	Identify

5	Generation of Accurate Alarm	Generate	Generate indistinct alarm
6	Positive rate	No false-positive rate	The massive number of the false-positive rate
7	Design	Simple	Complicated

Table- 2.5: Machine Learning Methods for Network Intrusion Detection

Learning Methods	Definition
Neural Networks (NNs)	Based on modeling, neurons operate in the human brain. To enhance neuron weights to minimize the error between actual and predicted training samples, single or multiple layers “perceptron” are trained with a gradient descent algorithm like Back Propagation.
Support Vector Machine (SVM)	Based on statistical learning theory, the Support Vector Machine depends on solving a quadratic optimization problem that calculates a linear separating hyperplane to create a classifier by maximizing the margin of separation between classes. It utilizes kernel mapping functions to perform calculations on the input space rather than the high dimensional feature space. Because only inner products are calculated, SVMs can learn a larger set of patterns and scales. This is because the classification complexity does not depend on the dimensionality of the feature space. The main disadvantage of the Support Vector Machine is that it can only handle binary-class classification problems whereas intrusion detection requires multiclass classification problems.

Source: Patel *et al.*, 2012.

2.13 Architecture-Based Intrusion Detection System

Further, Sayar *et al.* (2014), discussed Architecture Based Intrusion Detection System that can be categorized into two types such as,

2.13.1 Distributed

The Intrusion Detection System can be either distributed or centralized where the IDS numbers are distributed on the network while communicating with each other. They may also concentrate on a centrally located server. Further, the distributed systems extend protection to both host and network-based operations.

2.13.2 Passive and Reactive

These are the ingredients of the Behavior-based Intrusion Detection System where, while the active Intrusion Detection System function as a detector and preventer of intrusion, the reactive Intrusion Detection System solely concentrates on the detection of intrusions and hence, the reactive Intrusion Detection System is also identified as Intrusion Detection and Prevention System. (Sayer *et al.,* 2014).

2.14 Intrusion Prevention System (IPS)

Practically, an Intrusion Prevention System (IPS) which is software, or a hardware device has become indispensable in a network environment as it constitutes all the essential parameters of an Intrusion Detection System. It not only detects the known or unknown attacks in the network surroundings but also forbids intrusion for all possible incidents in the domain. In many ways, the IPS entertains the identified threats. Some of the prominent five categories as discussed by (Carr *et al.,* 2010) for such preventions at different layers which, however, firewalls are not able to decrypt as the attacks flow in an encrypted form are discussed below.

- Inline Network Intrusion Detection System;
- Application-based firewalls/ Intrusion Detection System;
- Layer seven switches;
- Network-based Application Intrusion Detection System and
- Deceptive Applications.

However, other mechanisms as mentioned below also can be applied to prevent such attacks in the network surroundings (Patel *et al.,* 2010).

- By reconfiguration other security controls in the system such as Firewall, or Router for anticipated and prospects attacks;
- Removal of malicious contents in network traffic by filtering out the imperiling packets;
- Configuring or reconfiguring other security and privacy controls in browser settings to prevent the anticipated attacks.

Mention may be made that, both IPS and IDS perform the detection of malevolent traffic, but they differ by type of response. While IDS warn of suspicious activity, IPS, a more advanced version of the technology is more active to prevent attacks in the system including other security solutions that react in real-time to block or prevent those activities. Consequent upon the increasing amount of use of sophisticated computing tools resulted to increase of malicious intrusion and therefore, designing of security measures have become crucial to prevent the same to protect the system resources and data including detecting the redundant attacks to facilitate appropriate action to repair the technological damages of the system.

2.15 Artificial Neural Network in Network Intrusion Detection System

From the foregoing discussions, it could be ascertained that Intrusion Detection System dynamically tracks processes in a monitored environment and it is employed for searching for any indication of a threat. Artificial Neural Networks are the mathematical models or computational models that are projected for stimulating specific organic brain function like pattern organization and it consists of many similar building blocks. Consisting of an interconnected group of artificial neurons, Artificial Neural Network is also known as Neural Network processes information using a connectionist approach to computation. Mention may be made that, connectionism relates to a set of various approaches in Artificial Intelligence, Cognitive Psychology, Cognitive Science, Neuroscience, and Philosophy of Mind which models' mental behavior phenomena as the emergent processes of interconnected networks of simple units. Artificial Neural Networks can identify and learn correlated patterns between input data sets and corresponding target values. After training, an Artificial Neural Network can be used to predict the outcome of new independent input data. It is appropriate to distinguish three types of units or layers such as (a) Input Layer, (b) Hidden Layer, and (c) Output Layer. (Dias *et al.*, 2017; Anifowose and Eludiora, 2012). The diagrammatical representation of the three layers of the Neural Network is placed below in Figure- 2.8.

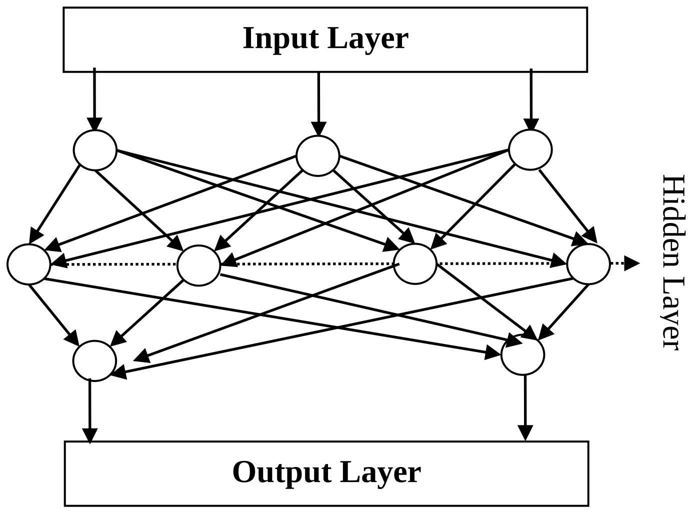


Fig.-2.8: Three layers of Neural Network

While explaining the function of three layers of Neural Network, while, the first layer consists of input neurons, the second layer comprises hidden neurons and the third layer contains output neurons. While, the supervised neural networks are trained to produce desired outputs in response to a training set of inputs, the unsupervised neural networks, on the other hand, are trained by letting the network continually adjusting itself to new input. Further, the supervised neural network is trained by providing input and matching output patterns and it is used in the modeling and controlling of dynamic systems, classifying noisy data, and predicting future events. The unsupervised neural networks are self-organized in which an (output) unit is trained to respond to clusters of patterns within the inputs. Reinforcement Learning is being considered as an intermediate form of the above two types of learning. Here the learning machine performs some actions on the environment and gets a feedback response from the environment. (Dias *et al.,* 2017).

- **Divisions and Activation Functions of Artificial Neural Network**

Artificial Neural Network (ANN) which is also known as Neural Network (NN) comprises a computational model or mathematical model which emulates the structure and functional aspects of Biological Neural Networks (Sammany *et al.*, 2007). Artificial Neural Network can be classified under three broad headings such as i) Network Architecture, ii) Learning methods and iii) Training modes. Further, a detailed division and the activation functions of each division of Artificial Neural Network are explained in Figure- 2.9

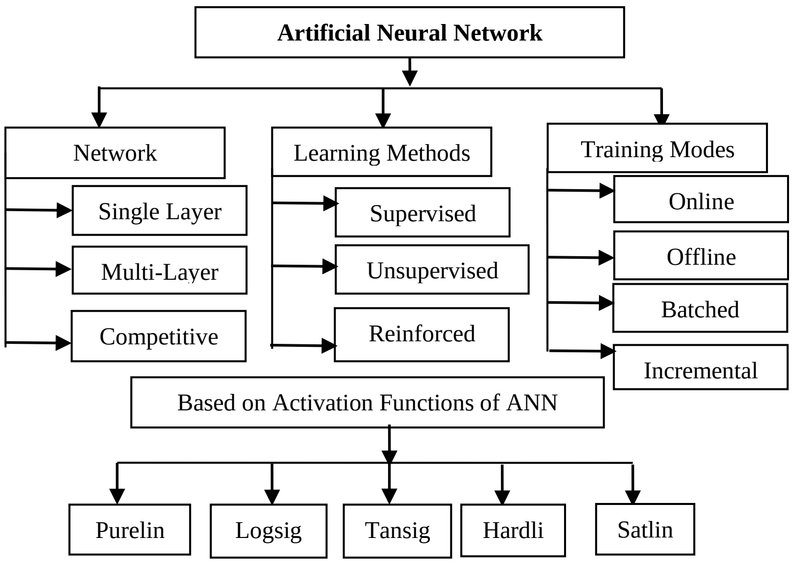


Fig.2.9: Divisions and Activation Functions of Artificial Neural Network
Source: Padhy and Simon, 2015

The descriptions about the different activation functions associated with Artificial Neural Network are as follows;

(i)	Purelin	-	Pure linear neural transfer Function.
(ii)	Satlin	-	Saturating linear transfer function.
(iii)	Logsig	-	Log-Sigmoid transfer function.
(iv)	Tansig	-	Tan-Sigmoid transfer function.
(v)	Hardli	-	Hard linear transfer function.

2.16 Learning Method

The learning method is otherwise known as a learning rule that is repeatedly applied over the network and it signifies a method or a mathematical logic that improves the performance of the artificial neural network. It is done by updating the weights and bias levels of a network when a network is simulated in a specific data environment.

The learning in neural networks can be referred to as information that moves between the neurons. Electrical stimulation along the dendrites is the form through which its' movement is being performed. Neuron generates an output to all other connected neurons once it receives some amount of simulation and thereafter, information takes its course to reach the destination where some reaction occurs. But, in case the incoming stimulation is too low then, no output is generated by the neuron and the information transfer will be obstructed. During the learning process, the connection structure among the neurons is changed. It means that there is a connection between the neural cells that once have learned a fact which enables the fast recall of this information. If some related information is acquired subsequently then the same neural cellos are stimulated and adapt their connection structure according to this new information.

Neural networks simulate the human brain's ability to learn. That is, the artificial neural network is also made of neurons and dendrites. Unlike the biological model, a neural network has an unchangeable structure that is built on a specified number of neurons and a specified number of connections between them (*"weights"*), which have certain values. Compared to the original, this means that incoming information "simulates" (exceeds a specified threshold value) certain neurons that pass the information to connected neurons or prevent further transportation along with the weighted connections. The value of weight is increased if the information is transported and decreased if not.

While learning different inputs, the values of the weights are changed dynamically until their values are *balanced*, so each input will lead to the desired output. The training of the neural network results in a matrix that holds the weight values between the neurons. When a neural network is trained correctly, it finds the desired output to a given input that had learning by using these matrix values.

The network gets ready for training or learning when for a specific application, it is structured. It operates by choosing randomly the initial weights at the beginning which follows the training or learning process. The training or learning paradigms of the network are grouped into (i) Supervised Learning, (ii) Unsupervised Learning, and (iii) Reinforcement Learning (Krenker *et al.*, 2011).

2.16.1 Supervised Learning

Supervised learning happens to be a machine-oriented learning technique where parameters of an Artificial Neural Network are set from training data which consists of pairs of both input and desired output value which are represented in data vectors. During the training, each input pattern is presented and propagated through the network to produce an output. There lies the difference between the actual and desired output unless the network is trained properly. The set of data that enables the training is known as the 'training set'. The same set of data is processed several times because the connection weights are refined (Krenker *et al.*, 2011).

2.16.2 Unsupervised Learning

Grippo (2000) viewed that while, for supervised learning presence of a teacher is required to specify the target output for every input, in unsupervised learning, no teacher is required. Here, the training data is unlabelled and untargeted. The system adapts to regularise data according to the rule implicit in its design. (Eskin *et al.*, 2002). The unsupervised learning modules are occasionally used as a constituent of a supervised learning structure. To be functional in supervised learning, a model must separate the data in a way that conserves the information needed for supervised learning.

2.16.3 Reinforcement Learning

Reinforcement learning is a machine learning procedure that sets parameters of an Artificial Neural Network where data is generally not given but is generated by connections with the situation. Reinforcement learning is concerned with how an artificial neural network must take action in an environment to maximize some conception of long-term reward. Reinforcement learning is regularly used as a part of an artificial neural network's overall learning algorithm. Again, Reinforcement learning is a more sensible model of low-level learning in human beings and animals. Reinforcement learning is predominantly suitable for problems that include a long-term against short-term reward trade-off. It has been applied effectively to diverse problems, including robot control, telecommunications, and games such as chess and other sequential decision-making tasks. (Krenker *et al.*, 2011).

2.17 Applications of Neural Network

The Neural Network is widely applied specially to solve problems that cannot be solved using conventional algorithms (Zhou, 2004) and such problems include generally optimization or classification. The Neural Network can be employed the various problem domains like,

- Pattern Association;
- Pattern Classification;
- Regularity Detection;
- Image Processing;
- Speech Analysis;
- Optimization;
- Robot Steering;
- Processing of inaccurate or incomplete inputs;
- Quality Assurance;
- Forecasting of Stock Market and
- Simulation etc.

There are different types of neural networks constituting different properties and each problem domain has its network type. Hence, it is flexible to solve different problems. The error tolerance is one of its ability where a neural network is trained for a specific problem to be able to recall correct results even if the problem which needs to be solved is not the same as the already learned one. While citing an example of this, during the learning process, some persons use to pronounce some words which are learned by the network and if trained properly, the neural network also recognizes the same words spoken by others. Hence, although, neural networks find solutions to the aforesaid problems, still confirmed or guaranteed results cannot be expected, and the results are the approximations of desired solutions with certain errors. Therefore, Theodorios and Koutroumbas (1999) viewed that, pattern recognition as the solution to such problems.

Haykin (1999) deduced that neural networks serve two functions such as (i) Pattern Classifiers and (ii) Nonlinear Adaptive Filters. Artificial Neural Network being adaptive, generally nonlinear system learns to perform a function (an input/output map) from data. Mention may be made that, adaptive means, the system parameters are changed during operation normally are known as the training phase. The other parameters of the Artificial Neural Network are fixed after the training phase and the system is deployed to solve the problems. Artificial Neural Network is built with a scientific procedure to optimize a performance criterion or to follow some implicit internal constraints commonly known as learning rule. It may be discussed that learning is the process by which the free parameters of a neural network get adapted through a process of stimulation by the environment in which the network is embedded and the type of learning is determined by the way the parameter changes. The set of well-defined rules for the solution of a learning problem is known as a learning algorithm and each learning algorithm is different from one another in the way in which the adjustment to the synaptic weight of a neuron is formulated (Sivanandam *et al.*, 2006).

The input/output data training data are fundamental in neural network technology as the necessary information is conveyed to discover the optimal operating point. The nonlinear nature of the neural network processing elements provides the system with elasticity for achieving the desired input/output map which may otherwise be expressed that, some Artificial Neural Networks are universal mappers.

Input is presented to the neural network along with a corresponding target response set at the output and in such case, the training is known as supervised. The difference between the desired response and the system output precipitates an error. This error information is fed back to the system and systematically adjusts the system parameters i.e., the learning rule, and this process continues until the acceptance of the performance. Thus, performance hinges excessively on the data. In the event of the non-availability of a significant portion of the data for the operating condition, the neural network cannot solve the problem. However, the neural network technology is the right platform to derive an approximate model for the good quantum of data and the problem of poorly understood. In Artificial Neural Networks, the designer selects the network topology, the performance function, the learning rule, and the criterion to stop the training phase but the system automatically adjusts the parameters. Hence, it is not easy to set prior information into the design and also hard to incrementally refine the solution especially when the system does not work properly. But, Artificial Neural Network-based solutions are not only befitting for the development time and resources but also providing the performance to the difficult problems. (Ahmed *et al.*, 2007).

2.18 Advantages and Disadvantages of Artificial Neural Network in Intrusion Detection System

Invariably, every detection system has both advantages and disadvantages depending on the architect's ability to use the technology potentiality. Artificial Neural Network has the following advantages and disadvantages (Pervez *et al.*, 2006).

● **Advantages**

i. Artificial Neural Network provides a flexible environment for any network to detect intrusion.

ii. The neural network is capable of analyzing any type of incomplete or distorted data from the network.

iii. The neural network possesses the ability to conduct an analysis of data in a non-linear fashion.

iv. It also has the power to process data from several sources in a non-linear fashion against the network in a coordinated attack by multiple attackers.

v. The intrinsic speed of Neural Networks is an added advantage in intrusion detection as the output of a Neural Network is expressed in the form of a probability.

vi. It also provides a predictive capability for misuse detection.

vii. A Neural Network-based misuse detection system identifies the chance that an event, or series of events, was indicative of an attack against the system.

viii. Neural Network in misuse detection can "learn" the characteristics of misuse attacks and identify instances that have been observed before by the network.

ix. The probability of an attack against the system also can be estimated and a potential threat flagged whenever the probability exceeds a specified threshold.

● **Disadvantages**

i. Neural Networks relate to the training requirements and hence, it is not applied to misuse detection.

ii. The ability of ANN which recognizes indications of an intrusion dependent on the accurate training data and the training methods of the system.

iii. The training routine requires a huge quantity of data to guarantee that the results are statistically accurate. The training of a Neural Network for misuse detection purposes may need thousands of individual attack sequences and this amount of responsive information is hard to achieve.

iv. The most important negative aspect of applying Neural Networks to intrusion detection is the ‘black box’ nature. The ‘Black Box’ is surrounded by hostile forces in Neural Networks in many applications.

2.19 Experiments of Different Algorithms of ANN using KDD dataset

The following neural network algorithms from the family of various neural network algorithms, based on their evaluation performance, the top 6 (six) best supervised neural network algorithms are selected and employed. A set of neural network algorithms comprise such as Learning Vector Quantization (LVQ), Radial Basis Function Neural Network (RBFN), Decremental Radial Basis Function Neural Network (DECR-RBFN), Evolutionary Radial Basis Function Neural Networks (EVRBFN), Multilayer Perceptron with Back-propagation Training (MLP-BP), Self-Optimizing Neural Networks (SONN). All the above algorithms are evaluated and compared the performance using KDD dataset. The detailed theoretical descriptions of all the neural network algorithms are described below (Nettleton *et al.,* 2010).

Artificial Neural Network comprises many networks out of which, for the present work six types of classifiers as follows have been chosen for the experiment based on their performances and accuracy for the Intrusion Detection System database.

- **Learning Vector Quantization (LVQ):**

Learning Vector Quantization is not only a prototype-based supervised classification algorithm but also the supervised counterpart of vector quantization systems. The basic LVQ algorithm is quite simple which starts from a trained Self-Organising Map (SOM) with input vectors {x} and weights/Voronoi vectors {w} (Bezdek, and Kuncheva, 2001).

- **Radial Basis Function Neural Network (RBFN):**

An RBF Neural Network consists of three layers, namely (i) Input layer, (ii) Hidden layer, and (iii) Output layer. For the neural network, the input layer takes care of initiating the input vector to each unit in the hidden layer. This is followed by the generation by each unit in the hidden layer and activation using the radial basis function used in the layer. The output is entirely based on the use of the activation function used in the hidden layer and the weights associated with the links between the hidden layer and the output layer (Broomhead and Lowe, 1988 & Howlett and Lakhmi, 2001).

- **Decremental Radial Basis Function Neural Network (DECR-RBFN):**

  The network (Broomhead and Lowe, 1988) output depends on the values yielded by every hidden neuron as well as the weight of the link connecting every hidden neuron with the output neuron. The output of every hidden neuron is the output of its function activation, and this function depends on the center of the neuron and a parameter call radius. Thus, the value provided by the neuron changes depending on the distance from the system input to the neuron center, and according to the radius value.

- **Evolutionary Radial Basis Function Neural Networks (EVRBFN):**

  EVRBFN is applied in Neural Network to build RBF NNs for optimizing their generalization error by finding the number of neurons in the hidden layer, and their centers and radius. The evolutionary algorithm itself has been programmed using the new evolutionary computation framework. It can be found at (https:// eodev. sourceforge.net) and is available under an open-source license. In this sense, every data structure one can imagine can be converted into something evolvable, and consequently, something that EO can evolve, optimizing it (Rivas *et al.*, 2004).

- **Multilayer Perceptron with Back-propagation Training (MLP-BP):**

  A multilayer perceptron (MLP) is a feed-forward artificial neural network model that maps many sets of input data into a set of appropriate outputs. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. MLP utilizes a supervised learning technique called back-propagation for training the network. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable (Rojas and Feldman, 1996).

- **Self-Optimizing Neural Networks (SONN):**

Kohonen's networks are one of the basic types of self-organizing neural networks. The ability to self-organize provides new possibilities for adaptation to formerly unknown input data. Kohonen's networks are a synonym of the whole group of nets that make use of self-organizing, competitive type learning method. There are many sub-types based on rivalry, which differ themselves by the precise self-organizing algorithm (Smotroff *et al.*, 1991).

2.20 Datasets used for Experiment

In this study, KDD Cup 1999 (Tavallaee *et al.*, 2009) intrusion dataset is used to evaluate each classification algorithm of Artificial Neural Network.

2.20.1 KDD’99 Cup dataset

KDD’99 dataset which originally hails from the DARPA’98 dataset comprises around 4,900,000 single connection vectors where every 41 features constitute and are labeled as normal or an attack with one specific attack type. This dataset is constructed and developed by Stolfo *et al.* (2000), based on the data captured in the DARPA’98 Intrusion Detection System Evaluation Program (Lippmann *et al.*, 2000) which was formerly used extensively as a benchmark dataset in the field of network intrusion detection system studies. The dataset contains a TCP-dump raw data of about 5 million connections collected from 7 weeks of network traffic records of training sets and 2 weeks records of test set data having around 2 million network traffic records. The data distribution of KDD’99 has discussed in Figure 1.4 of Chapter-1.

2.20.2 Experimental Setup

The whole neural network experiments are done by using Knowledge Extraction based on Evolutionary Learning (KEEL), which is an open-source (GPLv3) Java software available at (http://sci2s.ugr.es/keel/datasets.php) and it is used for a large number of different knowledge data discovery tasks (http://www.keel.es/). We used Windows 7 Home Basic (64-bit) as the testbed operating system, Intel(R) Core (TM) i3 CPU M 370 @ 2.40GHz processor, 3GB of

RAM. To test different classification algorithms of Artificial Neural Network, both the original KDD’99 is used for the whole experiments consisting of 41 features as well as 25190 training instances. All selected 6 different classifiers from various classification techniques of Artificial Neural Network were tested based on the k-fold cross-validation (K-FCV) technique with each dataset. K-FCV is one of the most common methods, where the dataset gets divided into k, (where k represents the no. of folds or subsets), k-1 subsets are used as training sets, and k-(k-1) subset is used for the testing set. In this study, a 10-fold cross-validation technique is applied, where 9 folds are used for training purposes and 1-fold is used for test purposes for comparison between various Artificial Neural Networks algorithms to know the best performance for Intrusion Detection System.

The parameters used for the experimental setup are mentioned below in Table- 2.6

Table 2.6: Parameter Description for Artificial Neural Network Experiments

1	**Learning Vector Quantization [LVQ]**	
	Parameter Descriptor	**Value**
	Iterations	100
	Neurons	20
	Neurons	0.3
	Nu	0.8
2	**Radial Basis Function Neural Network [RBFN]**	
	Parameter Descriptor	**Value**
	Neurons	50
3	**Decremental Radial basis Function Neural Network [DECR-RBFN]**	
	Parameter Descriptor	**Value**
	Percent	0.1
	nNeuronslni	20
	Alfa	0.3
4	**Evolutionary Radial Basis Function Neural Networks [EVRBFN]**	
	Parameter Descriptor	**Value**

	Neuronsrate	0.1
	Validationrate	0.15
	Popsize	100
	Tournamentsize	2
	Replacementrate	0.1
	Maxgenerations	100
	XoverRate	0.9
	mutator rate	0.1
5	**Multilayer Perceptron with Backpropagation Training [MLP-BP]**	
	Parameter Descriptor	**Value**
	hidden_layers	2
	hidden_ nodes	15
	Transfer	Htan
	Eta	0.15
	Alpha	0.10
	Lambda	10
	test_data	True
	validation- data	False
	cross- validation	False
	Cycles	10000
	Improve	0.01
	Problem	Classification
	tipify_inputs	True
	Verbose	False
	Saveall	False
6	**Self Optimizing Neural Networks [SONN]**	
	Parameter Descriptor	**Value**
	cross_validation	False
	Test_data	True
	tipify_inputs	True

	Tend	0.1
	Omega	5
	Maxnodes	1000
	To	1.0
	Alpha	0.99
	a_range	5.0
	LM_Convergence	0.0001
	Error	Missclass
	w_mse	1.0
	w_k	0.0

2.21 Description of Different Performance Matrices used for Experiment

One common method of determining the performance of a classifier is performed through using a confusion matrix (Kohavi and Provost, 1998). The performance of the different classifiers is monitored and measured by using various types of performance matrices such as i) Overall Accuracy, ii) Specificity, iii) Sensitivity, iv) G-Mean (Kubat *et al.*, 1998), v) Precision, vi) Recall, vii) F-Measure (Lewis and Gale, 1994) and viii) Matthews's Correlation Coefficient (MCC). A brief description of various performance matrices used for the study is mentioned below.

- **Overall Accuracy**

Overall Accuracy of a classifier indicates how well the classifier classifies the training instance. This means that the classifier correctly classifies the positive and negative classes without any misclassification.

- **Specificity or TNR**

Specificity is also known as True Negative Rate (TNR) which means that the proportions of negative classes in a binary classification test are correctly identified. It is also known as 'Inverse Recall' and it is the proportions of real negative cases that are correctly predicted negative. (Powers, 2007)

- **Sensitivity or TPR**

  The sensitivity indicates True Positive Rate (TPR) which means that the proportions of positive classes in a binary classification test are correctly identified. It is the proportion of Real Positives cases that are accurately predicted positive. This measures the coverage of the Real Positives cases by the +p (predicted positive) rule.

- **G-Mean**

  Geometric Mean (G-Mean) is also known as G-measure. It is a metric that is applied to evaluate the performance results by using both specificity and sensitivity. It ranges from 0 to 1 and an attribute that is perfectly correlated to the class provides a value of 1.

- **Precision or PPV**

  Precision is also called Positive Prediction Value (PPV) and it denotes the percentage of relevant objects that are identified for retrieval. The precision of a class is the number of true positives. It also denotes the amount of Predicted Positive cases or accurately Real Positives and is associated with the +P row.

- **Recall**

  The recall is defined as the number of true positives divided by the total number of elements that truly belong to the positive class. The recall is also known as sensitivity is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Recall associates only to the +R column.

- **F-Measure**

  The F-measure is also known as an F1 score or F-score is a measure of a test's accuracy and is defined as the weighted harmonic mean or average of both precision and recall of the test, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. A high F-Measure value ensures that both recall and precision are reasonably high. F-measure successfully references the True Positives to the Arithmetic Mean of Predicted Positives and Real Positives (Ingre and Yadav, 2015).

- **Matthews's Correlation Coefficient (MCC)**

Matthews's correlation coefficient (Matthews, 1975) is used in machine learning as a measure of the quality of binary (two-class) classifications. The MCC is, in essence, a correlation coefficient between the observed and predicted binary classifications, which returns a value between '−1' and '+1'. A coefficient of '+1' represents an ideal prediction, 0 no improved than a random prediction, and '−1' indicates the whole difference between prediction and observation.

2.21.1 Mathematical Formulae for all Performance Matrices

The mathematical formulae for all the performance matrices used to measure the performance of the different classifiers are placed below in Table- 2.7.

Table 2.7: Mathematical Formulae for all the Performance Matrices

$Overall\ Accuracy\ (OV) = \frac{TP+TN}{TP+FP+FN+TN}$	(1)
$Specificity = \frac{TN}{TN+FP}$	(2)
$Sensitivity = \frac{TP}{TP+FN}$	(3)
$G - Mean = \sqrt{Specificity * Sensitivity}$	(4)
$Precision = \frac{TP}{TP+FP}$	(5)
$Recall = \frac{TP}{TP+FN}$	(6)
$F = 2.\frac{Precision \cdot Recall}{Precision + Recall}$	(7)
$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$	(8)

Sources: Fawcett (2006), Powers (2011), and Ting (2011)

Where,	TP = Total number of correctly classify positive examples
	FP = Total number of miss-classified negative examples
	TN = Total number of correctly classify negative examples
	FN = Total number of miss-classified positive examples

2.22 KEEL-Dataset Repository- An Introduction

KEEL (Knowledge Extraction based on Evolutionary Learning) 3.0 is an open-source (GPLv3) Java software that was originally developed in 2009 and later upgraded in 2009 as a tool concerning the implementation of evolutionary algorithms and soft computing techniques for standard DM problems such as regression, classification or association rules, as well as data pre-processing technique. It is used for a large number of different knowledge data discovery tasks and provides a simple GUI based on a data flow to design experiments with different datasets and computational intelligence algorithms to assess the behaviour of the algorithms. It contains a wide variety of classical knowledge extraction algorithms, preprocessing techniques (training set selection, feature selection, discretization, and imputation methods for missing values, among others) computational intelligence-based learning algorithms, hybrid models, statistical methodologies for contrasting experiments, and so forth. It allows performing a complete analysis of new computational intelligence proposals in comparison to existing ones. The primary objectives of the KEEL concentrate on research and education (http://www.keel.es/). A package named RKEEL 32 has been recently created as a layer between R and KEEL, allowing users to execute KEEL functionalities from R (Triguero *et al.*, 2017).

To highlight a brief account, the KEEL Suite 3.0 comprises five different blocks such as i) Data Management, ii) Experiments, iii) Educational, iv) Modules and v) Help.

- **Data Management**

  The primary features attached to this component are

i) Import Data: Here, the user can export the data from KEEL format files to other formats;

ii) Export Data: The user is allowed to import data from other format files to the KEEL format.

iii) Visualize Data: The user can visualize the existing KEEL format datasets.

iv) Edit Data: The user gets the option to edit existing KEEL format datasets.

v) Make Partitions: The user can make partitions for existing KEEL datasets (http:// sci2s.ugr.es/keel/documents/KeelManual3.0.pdf).

- **Design of Experiment (Off-line Module)**

The objective of the design of the experiment is to allow a user to create the desired experiments using a graphical interface over the selected datasets. To perform this, the user can use the selected datasets and algorithms to generate a file containing a folder structure with all the necessary required files to run the designed experiments in the processing unit (http://sci2s.ugr.es/keel/documents/ KeelManual 3.0.pdf).

- **Educational Experiments**

This allows for the design of experiments that can be run step-by-step to display the learning process of a certain model by using the software tool for educational purposes (Alcalá-Fdez *et al.*, 2011).

- **Modules**

KEEL software has several modules that are meant for particular purposes. Specifically, three different modules have been developed. The type of modules includes, i) Imbalanced Learning Module, ii) Statistical Test Module, iii) Semi-supervised Learning Module and iv) Multiple Instance Learning Module. (http://sci2s.ugr.es/keel/ documents/ Keel Manual3 .0. pdf).

2.23 Proposed Artificial Neural Network Experimental Design

For the experiments of various neural network algorithms, an experimental framework has been designed in Fig.-2.10. The proposed framework consists of network traffic data, data pre-process, various neural network algorithms, and different performance matrices are used to measure the performance and the details are described in Table-2.7 above.

- **Data pre-process**

The network traffic dataset was initially prepared, analyzed, and converted from multiclass to binary by combining all the attack types and to make it normal and anomaly. After pre-processing and reducing redundant data, 25190 instances are selected for the experimentation dataset. For the whole experimentation, full 41 features are obtained from the KDD dataset. A detailed description of the dataset used for training and testing experiments has discoursed in section 2.20.

- **Experimental Framework**

In this experiment, the KDD dataset is taken as input for the different algorithms i.e., LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN. To evaluate the performance of different algorithms, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and Matthews's Correlation Coefficient are used. The architecture of the experimental framework is placed below in Figure-2.10.

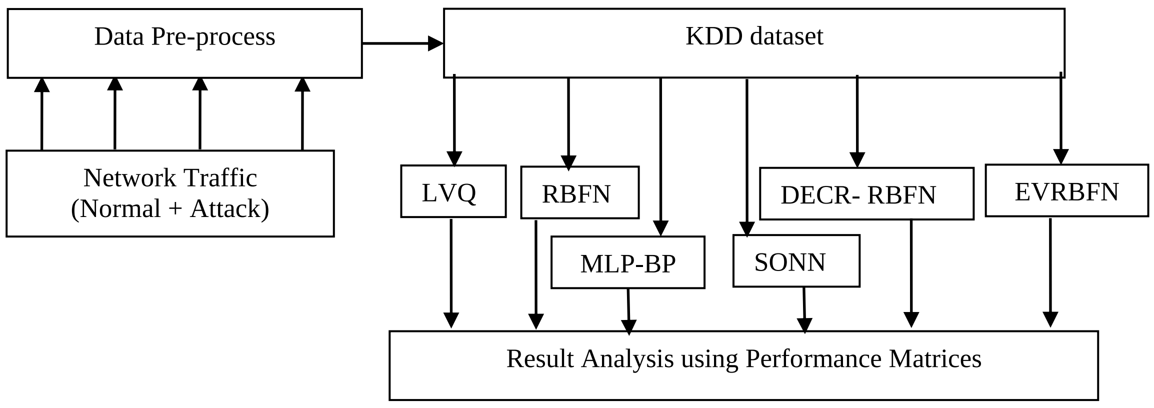


Fig. 2.10: Experimental Framework of Artificial Neural Network

2.24 KDD dataset for Artificial Neural Network Experiment (training and test)

The KDD (training and test) dataset are analyzed after pre-processing and reducing redundant data, 25190 input instances are selected for experimentation purposes. Then trainset was divided into ten sets randomly, containing both normal and attack data that appears in the KDD dataset.

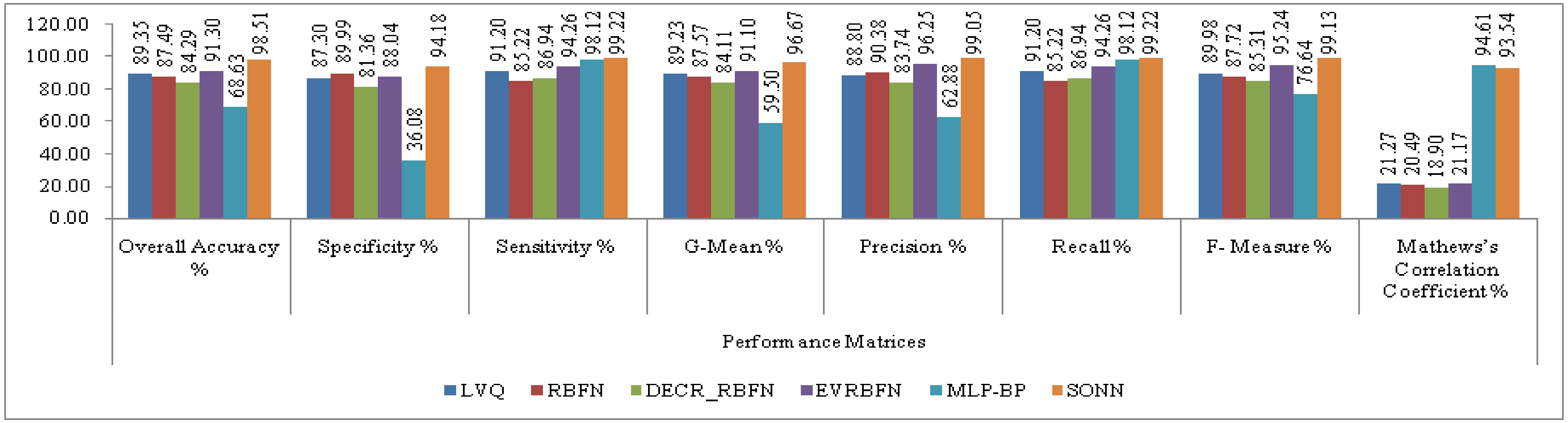
The different attacks contained in the KDD dataset are back, land, Neptune, pod, smurf, teardrop, satan, ipsweep, nmap, portsweep, guess_password, ftp_write, imap, phf, multihop, warezmaster, warezclient, spy, buffer_overflow, load-module, perl, and rootkit, etc. These all-attack types fall under four attack categories i.e., Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probe. Out of ten datasets, nine datasets were selected for training and one dataset was selected randomly for the test. The training datasets comprise total instances, 929511 (2519*41* 9=929511) while, the test dataset constitutes the total instances, 103279 (2519*41*1= 103279). The program itself randomly selects the data for training and testing purposes. The distribution of data for the KDD dataset is shown in Table-2.8.

Table 2.8- Distribution of data for KDD (training & test) experiments

Dataset Name	No. of Features	Input instances	Total
Trainset data 1	41	2519	2519* 41*1= 103279
Trainset data 2	41	2519	2519* 41*1= 103279
Trainset data 3	41	2519	2519* 41*1= 103279
Trainset data 4	41	2519	2519* 41*1= 103279
Trainset data 5	41	2519	2519* 41*1= 103279
Trainset data 6	41	2519	2519* 41*1= 103279
Trainset data 7	41	2519	2519* 41*1= 103279
Trainset data 8	41	2519	2519* 41*1= 103279
Trainset data 9	41	2519	2519* 41*1= 103279
	Total Training instances		**929511**
Testset data 1	41	2519	2519 *41*1= 103279
	Total Test instances		**103279**

Table: 2.9: Experimental Results of Artificial Neural Network Algorithms using KDD (training) dataset

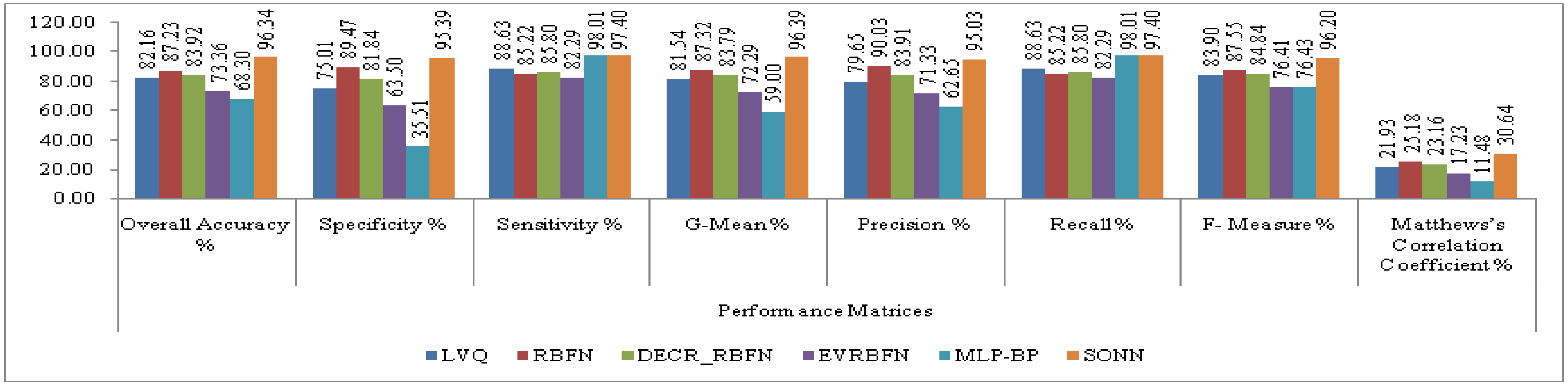
Algorithms	Performance Matrices							
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-measure %	Matthews's Correlation Coefficient %
LVQ	89.35	87.30	91.20	89.23	88.80	91.20	89.98	21.27
RBFN	87.49	89.99	85.22	87.57	90.38	85.22	87.72	20.49
DECR-RBFN	84.29	81.36	86.94	84.11	83.74	86.94	85.31	18.90
EVRBFN	91.30	88.04	94.26	91.10	96.25	94.26	97.09	21.17
MLP-BP	68.63	36.08	98.12	59.50	62.88	98.12	76.64	94.61
SONN	98.51	94.18	99.22	96.67	99.05	99.22	99.13	93.54



Graph-2.2: Experimental Results of Artificial Neural Network Algorithms using KDD (training) dataset

Table-2.10: Experimental Results of Artificial Neural Network using KDD (test) dataset

Algorithms	Performance Matrices							
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F- Measure %	Matthews's Correlation Coefficient %
LVQ	82.16	75.01	88.63	81.54	79.65	88.63	83.90	21.93
RBFN	87.23	89.47	85.22	87.32	90.03	85.22	87.55	25.18
DECR-RBFN	83.92	81.84	85.80	83.79	83.91	85.80	84.84	23.16
EVRBFN	73.36	63.50	82.29	72.29	71.33	82.29	76.41	17.23
MLP-BP	68.30	35.51	98.01	59.00	62.65	98.01	76.43	11.48
SONN	96.34	95.39	97.40	96.39	95.03	97.40	96.20	30.64



Graph.2.3 Experimental Results of Artificial Neural Network using KDD (Test) dataset

2.25 Experimental Results

In this section, the experimental results of six different Artificial Neural Network algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN are discussed. To evaluate the performance of these algorithms' different performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and Matthews's Correlation Coefficient are used. The evaluated results for both training and test were obtained after proper validation of the experiments.

2.25.1 Experimental Results of KDD (training) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews's Correlation Coefficient for six different neural network algorithms i.e, LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN using KDD (training) dataset as stated in section 2.20 is placed in Table- 2.9 with the corresponding Graph- 2.2.

The table shows that the SONN algorithm has performed the best result for Overall Accuracy i.e, 98.51%. It means that this algorithm accurately classifies the training dataset as compared to other networks. But, the algorithm EVRBFN classifies the instance with 91.30%, whereas other algorithms such as LVQ, RBFN, DECR-RBFN, and MLP-BP resulted in 89.35%, 87.49%, 84.29%, and 68.63% respectively while classifying the training dataset.

The result for Specificity or True Negative Rate, again the SONN algorithm performed the best result i.e, 94.18% which otherwise means that the SONN algorithm correctly classifies both positive and negative classes. But other algorithms like RBFN, LVQ, EVRBFN, and DECR-RBFN classify the dataset with 89.99%, 87.30%, 88.04%, and 81.36% respectively. But, the MLP-BP algorithm classifies the training instance with 36.08%, which shows very low performance as compared to other algorithms.

The result for Sensitivity or True Positive Rate, again SONN algorithm shows the best performance i.e. 99.22%. It means that the SONN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithms like MLP-BP and EVRBFN show 98.12% and 94.26% respectively and it is also considered as better performance for Sensitivity as compared with SONN. But the other algorithms like LVQ, DECR-RBFN, and RBFN and performed 91.20%, 86.94%, and 85.22% respectively which otherwise means that RBFN algorithm shows very low performance as it could not classify the training dataset correctly.

For the result of G-mean or G-measure, the SONN algorithm performed the best i.e. 96.67% among others, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (94.18%) and Sensitivity (99.22%) of SONN are high then the G-mean of SONN (96.67%) is also relatively high. Further, the G-mean value for EVRBFN constitutes 91.10% because the value of both Specificity (88.04%) and Sensitivity (94.26%) are also superior. But the performances of the other algorithms like LVQ (89.23%), RBFN (87.57%) and DECR-RBFN (84.11%) not very good. Finally, the performance of MLP-BP (59.50%) is very low as compared to other algorithms; because its Specificity value is much lower i.e. 36.08%.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 99.05% of exactness or quality of the classifier. But the other algorithms like EVRBFN (96.25%) of exactness, whereas RBFN (90.38%), LVQ (88.80%), DECR-RBFN (83.74%), and MLP-BP (62.88%) of exactness. Ultimately it deduced that the MLP-BP algorithm shows very low performance as compared with others.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate (TPR), again the SONN algorithm shows the best performance i.e. 99.22%. It means that the SONN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithms like MLP-BP and EVRBFN show 98.12% and 94.26% respectively and it is also considered as better performance for Sensitivity as compared with SONN. But the other algorithms like LVQ, DECR-RBFN, and RBFN and performed 91.20%, 86.94%, and 85.22% respectively which otherwise means that RBFN algorithm shows very low performance as it could not classify the training dataset correctly.

The result of the F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here the SONN algorithm gives the best result i.e. 99.13% because the result for both Precision and Recall is relatively very high. i.e. 99.05% and 99.22% respectively. Then the result for the EVRBFN algorithm shows 97.09% because the result for both Precision and Recall also shows very good i.e. 96.25% and 94.26%. Then the result of the LVQ algorithm comes with 89.98% because of the result of Precision i.e. 88.80%. Subsequently, for RBFN, DECR-RBFN and MLP-BP algorithms, the result of F-measure comes with 87.72%, 85.31%, and 76.64% respectively because the Recall value of RBFN algorithm is 85.22% whereas, the respective Precision value of both DECR-RBFN and MLP-BP is very low i.e. 83.74% and 62.88%.

The result of MCC for MLP-BP algorithm shows the best result among all other algorithms i.e. 94.61% whereas, SONN shows 93.54% for MCC. But the other algorithms like LVQ, EVRBFN, RBFN, DECR-RBFN algorithms show very low performance i.e. 21.27%, 21.17%, 20.49%, and 18.90% respectively.

2.25.2 Experimental Results of KDD (test) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews’s Correlation Coefficient for six different neural network algorithms i.e, LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN using KDD (test) dataset as stated in section 2.20 is placed in Table- 2.10 with the corresponding Graph- 2.3.

The table shows that the SONN algorithm has performed the best result for Overall Accuracy i.e. 96.34%. It means that this algorithm accurately classifies the test dataset as compared to other algorithms. But the algorithm, RBFN classifies the instance with 87.23%, whereas other algorithms such as DECR-RBFN, LVQ, EVRBFN, and MLP-BP algorithms resulted in 83.92%, 82.16%, 73.36%, and 68.30% respectively while classifying the test dataset.

The result for Specificity or True Negative Rate, again the SONN algorithm performed the best result i.e. 95.39% which otherwise means that the SONN algorithm correctly classifies both positive and negative classes. But other algorithms like RBFN classify the dataset with 89.47% whereas, DECR-RBFN, LVQ, and EVRBFN classify the dataset with 81.84%, 75.01%, and 63.50% respectively which shows very low performance. But, the MLP-BP algorithm classifies the test instance with 35.51%, which also reveals very low performance as compared to other algorithms it means that this algorithm could not classify the dataset accurately.

The result for Sensitivity or True Positive Rate, again MLP-BP algorithm shows the best performance i.e. 98.01%. It means that the MLP-BP algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithms like SONN and LVQ show 97.40% and 88.63% respectively and it is also considered as better performance for Sensitivity as compared with MLP-BP. But the other algorithms like DECR-RBFN, RBFN, and EVRBFN performed 85.80%, 85.22%, and 82.29% results respectively, which otherwise means that EVRBFN algorithm shows very low performance as it could not classify the test dataset correctly.

For the result of G-mean or G-measure, the SONN algorithm performed the best result i.e. 96.39% among others, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (95.39%) and Sensitivity (97.40%) of SONN are high. So, the G-mean of SONN (96.39%) is also reasonably high. Further, the G-mean value for RBFN constitutes 87.32% because the value of both Specificity (89.47%) and Sensitivity (85.22%) is also higher. But the performances of the other algorithms like DECR-RBFN, (83.79%), LVQ (81.54%), EVRBFN (72.29%) which is not very good. Finally, the performance of MLP-BP (59.00%), which is very low as compared to other algorithms, because its Specificity value is much lower i.e. 35.51%.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 95.03% of exactness or quality of the classifier. But the other algorithms like RBFN (90.03%) of exactness, whereas DECR-RBFN (83.91%), LVQ (79.65%), EVRBFN (71.33%), and MLP-BP (62.65%) of exactness. Ultimately it deduced that the MLP-BP algorithm shows very low performance as compared with other algorithms.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate (TPR), MLP-BP algorithm shows the best performance i.e. 98.01%. It means that the MLP-BP algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithms like SONN and LVQ show 97.40% and 88.63% respectively and it is also considered as better performance for Sensitivity as compared with MLP-BP. But the other algorithms like DECR-RBFN, RBFN, and EVRBFN performed 85.80%, 85.22%, and 82.29% results respectively, which otherwise means that EVRBFN algorithm shows very low performance as it could not classify the training dataset correctly.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here the SONN algorithm gives the best result i.e. 96.20% because the result for both Precision and Recall is relatively very high. i.e. 95.03% and 97.40% respectively. Then the result for the RBFN algorithm shows 87.55% because the result for both Precision and Recall also shows very good. i.e. 90.03% and 85.22%. Then the result of DECR-RBFN and LVQ algorithm comes to 84.84% and 83.90%. But, for MLP-BP and EVRBFN algorithms, the result of F-measure comes to 76.43% and 76.41%

respectively only because of their corresponding Precision value, which is very low i.e. 62.65% and 71.33%.

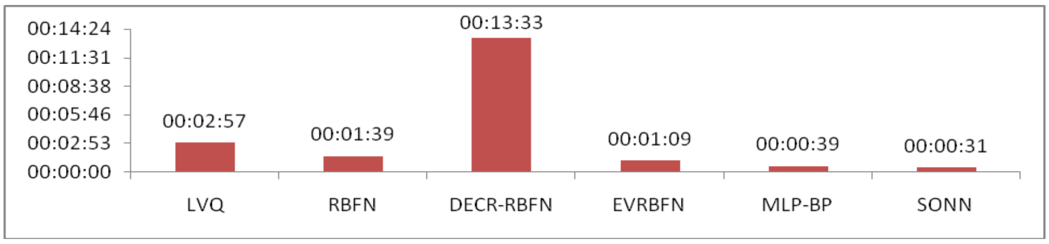
The result of MCC for the SONN algorithm shows the best result among all other algorithms i.e. 30.64% whereas, the RBFN algorithm shows 25.18% for MCC. But the other algorithms like DECR-RBFN, LVQ, EVRBFN, MLP-BP algorithms show very low performance i.e. 23.16%, 21.93%, 17.23%, and 11.48% respectively.

2.25.3 Training time for Experiment

The training time consumed for the experiments of the different Artificial Neural Network algorithms to train the KDD dataset is shown in Table-2.11 with the corresponding Graph-2.4. From this table, we deduced that the SONN algorithm is taking the optimal time i.e., 00:00:31 to train the dataset.

Table-2.11: Training time of the Artificial Neural Network Algorithms

Algorithms	Training Time
LVQ	00:02:57
RBFN	00:01:39
DECR-RBFN	00:13:33
EVRBFN	00:01:09
MLP-BP	00:00:39
SONN	00:00:31



Graph: 2.4: Training time of the Artificial Neural Network Algorithms

2.26 Support Vector Machine in the Intrusion Detection System

Boser *et al*. (1992) introduced the concept of Support Vector Machine in COLT-92. While tracing a brief genealogy of Support Vector Machine (SVM), it is a set of related supervised learning methods used for classification and regression and belongs to a family of generalized linear classifiers. In other words, a Support Vector Machine (SVM) is a classification and regression prediction tool that uses machine learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. Support Vector Machine can be defined as systems that use hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. The Support Vector Machine was initially familiar with the NIPS community and now is an active part of machine learning research around the world. SVM becomes well-known when using pixel maps as input; it gives accuracy comparable to sophisticated neural networks with elaborated features in a handwriting recognition task. It is also being used for many applications, such as handwriting analysis, face analysis, and so forth, especially for pattern classification and regression-based applications. The Support Vector Machine (SVM) which was developed by Vapnik (1995) gained acceptance due to many promising features for better empirical performance.

The formulation uses the Structural Risk Minimization (SRM) principle, which has been shown to be superior (Burges, 1998) to the traditional Empirical Risk Minimization (ERM) principle, used by conventional neural networks. SRM minimizes an upper bound on the expected risk, whereas ERM minimizes the error on the training data. It is this difference that equips SVM with a greater ability to generalize, which is the goal of statistical learning. SVMs were developed to solve the classification problem, but recently they have been extended to solve regression problems (Vapnik, 1997).

The classifiers of Support Vector Machine are originally designed for binary classification. Support Vector Machine is based on the idea of a hyperplane classifier, or linearly separability. In data mining, the Support Vector Machine is proved to be a successful classification algorithm. The application of Support Vector Machine in Data Mining and Bio-Informatics requires the processing of huge data sets for which the training time of SVM too much while processing such data sets. Even it takes years to train SVM on a data set consisting of one million records. Therefore, the proposals were mooted to enhance its training performance either through random selection or approximation of the marginal classifier. But, such approaches are still not feasible with large data sets where even multiple scans of the entire data set are too expensive to perform or result in the loss through over-simplification of any benefit to be gained through the use of Support Vector Machine. (Kashyap *et al.,* 2013).

Machine Learning is measured as a subfield of Artificial Intelligence and it is concerned with the growth of techniques and methods which facilitate the computer to learn. In simple terms expansion of algorithms enables the machine to learn and carry out tasks and performances. Machine learning overlaps with statistics in several ways. Over a while, many techniques and methodologies were developed for machine learning tasks (Burges, 1998). The necessary idea behind SVMs is that input vectors are mapped in a nonlinear method into a high measurement feature space determining the best possible hyperplane that separates data into classes such that the “margin” or distance between the members of diverse classes is maximized. (Cristianini and Taylor, 2000)

Support Vector Machine is one of the most accepted and popular techniques of typical pattern recognition tool which is widely used in intrusion detection. Support Vector Machine could be used in three different ways in the intrusion detection process.

(i) Support Vector Machine could be applied directly to ascertain the normal activities model of a computer system. To cite a few of them, while, Yao *et al*. (1997). applied rough sets enhanced Support Vector Machine model for intrusion detection on both sequence-based and feature-based datasets, Deng *et al*. (2003), applied Support Vector Machine based Intrusion Detection System for network layer security. Laskov *et al*. (2006), applied the Support Vector Machine for online learning, network traffic monitoring of industrial devices while, Zanni *et al*. (2006), introduced parallel software to solve the classification problems through quadratic program arising in training Support Vector Machine. Tian *et al*. (2007), applied the anomaly detection method on a sequence-based dataset using a one-class Support Vector Machine.

(ii) Use of both Support Vector Machine and neural network provide potential result in ranking the features of intrusion detection dataset (Sung and Mukkamala, 2003). They further viewed that, compared to neural networks, the Support Vector Machine not only could train with a larger number of patterns in less time but also the detection speed of Support Vector Machine is quite faster than the neural networks.

(iii) To solve the intrusion detection problems, the Support Vector Machine also could be applied with other Artificial Intelligence and software computing techniques. Chen *et al*. (2005), in the beginning, applied genetic algorithms to analyze and optimize the data and used the Support Vector Machine to classify the optimized data. Tsang *et al.* (2005), applied a computational geometry algorithm to increase the speed of Support Vector Machine on the good quantum of intrusion dataset by adopting a Support Vector Machine kernel function.

2.27 Support Vector Machine in Network Intrusion Detection System

Support Vector Machine is one of the effective application tools in a network intrusion detection system and could be effectively utilized in the KDD cup 1999 dataset. Most of the previous researches on Support Vector Machine concentrate on its applications intrusion detection and compared with neural networks and the results establish the fact that Support Vector Machine was preferred because of their apparent advantages over neural networks in terms of time consumption, speed, accuracy, efficiency, and scalability, etc. especially towards the use intrusion detection dataset. (Mukkamala *et al.,* 2002). Further explanations by the authors established that, although both Neural Networks and Support Vector Machine have high accuracy in intrusion detection still, they viewed that the Support Vector Machine could only classify the intrusion dataset accurately into two classes, “attack” or “normal”. Further studies revealed that, while comparing between the performances of various algorithms like Support Vector Machine, Neural Networks, Multivariate Adaptive Regression Splines (MARS), and Linear Genetic Programs (LGPs) it was found that Support Vector Machine performed accurately for both the training and testing time but less accurate than LGPs (Mukkamala and Sung, 2003a). One more study found that the performance of the Support Vector Machine solved the different problems regarding multiple classes rather than resolving the binary classification only (Ambwani, 2003) and feature selection (Mukkamala and Sung (2003b, c). The idea was to select only the important features from the intrusion dataset to improve only the training, testing, and accuracy of the intrusion detection system.

Further research focusing on the denial-of-service attacks by reducing the features in Support Vector Machine concluded that the accuracy was reduced. Again, to improve the accuracy of the dataset with reduced features, the authors used a small dataset and achieved a high detection rate (Mukkamala and Sung, 2003c; Mukkamala and Sung, 2005).

In 2005, various researchers continued the experiment with Artificial Neural Networks and Support Vector Machine began to attempt different approaches for the improvement towards the overall system performance for Intrusion Detection System. Chen *et al*. (2005), studied an approach generally used in text mining and applied it towards the intrusion detection problem where a text representation method uses the term frequency (*tf)* multiplied by its inverse document frequency *(idf)* i.e. (*tf* x *idf)* to conclude the uniqueness of each term and a transformation of this data and this was used to train a Support Vector Machine with an RBF kernel. The study found that Support Vector Machine performed perfectly compared to Artificial Neural Networks. But the results did not support the idea that the use of such heuristic approaches as frequency encoding would lead to an enhancement in detection performance. The researchers again studied intensively at the model selection and experimented with kernel parameters in conjunction with the continuation of their research on feature reduction where the result deduced that kernel parameter selection affected the accuracy along with feature reduction, without the description of the implementation method (Mukkamala *et al.,* 2005).

Likewise, Xu (2006) published a study using a Support Vector Machine with an RBF kernel method on about 0.2% of the KDD Cup 1999 dataset using a “one versus all” approach towards multiple classification problems. In this approach, binary datasets were generated for each class category and binary classifiers were trained on each of these datasets and a confidence value was calculated. During the classification period, each sample was fed into each of the binary classifiers and an ultimate determination of the class was made based on the confidence value which is highest for that sample (Witten *et al.,* 2011). Xu (2006) utilized principal components analysis to select the most significant data features and a “sequential pattern prediction” method to assist and determine whether a pattern was “normal” or “anomalous” based on a reinforcement learning approach. Although the results appear to be sound, the data set was small and the methodology used for the sequential pattern prediction was not adequately explained.

In 2007, the focus of the researchers again concentrated on Support Vector Machine in the area of network intrusion detection with hybrid system approaches. Peddabachigari *et al.* (2007), developed a hierarchical model and employed in the first stage a decision tree to extract or learn “rules” from the data and then processed with Support Vector Machine which resulted in excellent detection of probing attacks. Another approach initiated by Hwang *et al*. (2007), employed a three-tiered scheme made up of a blacklist to filter out known attacks, a whitelist recognized as normal traffic, and a Support Vector Machine. Then the system achieved excessive detection rates on the diverse traffic categories but did not provide any information about the false positive rate.

2.28 Statistical Learning Theory

The statistical learning theory provides a structure for studying the problem of gaining knowledge, making predictions, making decisions from a set of data. In easy terms, it enables the choosing of the hyperplane space in such a way that it closely represents the underlying function in the target space. (Evgenuiu, 1998).

In statistical learning theory, the problem of supervised learning is formulated as follows. We are given a set of training data $\{(\mathbf{x}_1,y_1)\ldots(\mathbf{x}_n,y_n)\}$ in $R^n \times R$ sampled according to unknown probability distribution $P(\mathbf{x},y)$ and a loss function $V(y,f(\mathbf{x}))$ that measures the error, for a given $\boldsymbol{x}$, $f(\boldsymbol{x})$ is "predicted" instead of the actual value y. The problem consists of finding a function 'f' that minimizes the expectation of the error on new data that is, finding a function 'f' that minimizes the expected error. $\int V(y,f(x))\, P(x,y)\, dx\, dy$. In statistical modeling, we would choose a model from the hypothesis space, which is closest (concerning some error measure) to the underlying function in the target space. (Evgenuiu, 1998). The basic knowledge approach utilized by Support Vector Machine is based on an optimal learning algorithm imitative from statistical learning theory. It has also a quadratic programming solution. The Support Vector Machine is a significant machine learning procedure developed for binary classification problems and afterward extended towards solving different regression problems. Based on statistical learning theory, Support Vector Machine map input vectors in a non-linear manner into a high dimensional feature space so that, an optimal separating hyperplane can be established. (Cortes and Vapnik, 1995).

Support Vector Machine is based on the idea of "maximizing the margin" among the classes in a set of training data. This idea was initially offered by Vapnik (1992) from Bell Laboratories. The author also introduced to design the idea of supporting patterns and suggested the mathematics for the training algorithm. (Boser *et al.*, 1992).

2.29 Advantages and Disadvantages of Support Vector Machine in Intrusion Detection Systems

- **Advantages**

The advantages of the Support Vector Machine technique can be summarised as follows. (Auria and Moro, 2008).

i. The use of the Support Vector Machine gained momentum due to its generalization in pattern classification using various kernel tricks.
ii. Support Vector Machine provides a platform to detect the intrusions in real-time very quickly and also provides a solution.
iii. It also updates the training patterns dynamically in case of a new pattern during classification (Jha and Ragha, 2013).
iv. It is a supportive and dynamic tool to detect network intrusion.
v. Support Vector Machine can avoid the unforeseen of very high dimensional representations in classification.
vi. The training of the dataset in the Support Vector Machine is relatively easy.
vii. It scales relatively well to high dimensional data and the trade-off between classifier complexity and error can be controlled explicitly (Burges, 1998; Lewis, 2004).

- **Disadvantages**

i. A major drawback of SVMs is that they cannot directly learn multiclass classification tasks and therefore must employ strategies combining results from multiple runs of binary classification trials such as "one against one" and "one against all" which all require extra processing resources. (Huang *et al.*, 2012).
ii. The application of kernel parameter selection affects the accuracy of the Support Vector Machine.
iii. Unlike conventional statistical and neural network methods, the SVM does not control model complexity by keeping the number of features small.
iv. Unlike Artificial Neural Networks, the computational complexity of Support Vector Machine does not depend on the dimensionality of the input space.

2.30 Conclusion

The exponential growth of the internet has opened new vistas in different operations, which create a viable environment for the development of society. It also simultaneously propagated the miscreants to make intrusions in various fields. Many of the authors and scientists have concentrated upon their studies in making out the intrusions in various fields. This has been a potential discussion to control the mechanism employed by the intruders.

A way-out methodology such as Artificial Neural Network and Support Vector Machine has been initiated to determine the various aspects of intrusions because a discussion on Intrusion Detection System has become paramount in the internet domain. There are many advantages and disadvantages also in Intrusion Detection System. However, compared to the disadvantages, the advantages are prolific.

The performance of various neural network classification algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN have been evaluated based on KDD dataset with full 41-dimension features was used throughout the experiments. To measure the performance of the different algorithms, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews's Correlation Coefficient (MCC) were used. Among various tested classification algorithms, the SONN algorithm shows the best result (Overall Accuracy) compared to other tested algorithms. According to the time constraints again SONN algorithm revealed the best result to train the dataset.

CHAPTER-3

EXPERIMENTS USING ARTIFICIAL NEURAL NETWORK (ANN)

Experiments Using Artificial Neural Network (ANN)

3.1 Introduction

Intrusion detection, a significant research problem in network security reckons monitoring processes and analysing network traffic data to find security infringements. The mining approach is one of the accepted podiums to develop an intrusion detection system as it leads to inspect network traffics and alerts the administrator in case of unauthorized access or attempts by a stranger. The network traffic can be classified into normal and anomalous to detect intrusions.

3.2 Algorithms

A set of top 6 (six) best supervised neural network algorithms based on their evaluation performance from the family of various neural network algorithms are selected and employed. The set of neural network algorithms comprise Learning Vector Quantization (LVQ), Radial Basis Function Neural Network (RBFN), Decremental Radial Basis Function Neural Network (DECR-RBFN), Evolutionary Radial Basis Function Neural Networks (EVRBFN), Multilayer Perceptron with Back-propagation Training (MLP-BP), Self-Optimizing Neural Networks (SONN). All the above algorithms are evaluated and compared the performance using NSL-KDD dataset. The detailed theoretical descriptions of all the neural network algorithms are described in Chapter-2, Section 2.15.

3.2.1 NSL-KDD dataset

Tavallaee *et al.* (2009), proposed the NSL-KDD dataset, which is an enhanced edition KDD Cup '99 dataset created by the DARPA at the MIT Lincoln Laboratories USA. The KDD'99 Cup dataset encompasses extensive records of pleonastic data, where 78% of training dataset and 75% of test dataset are found to be duplicates. This may direct the classifier algorithm unreasonable towards the other repeated records and therefore, avoid it against harmful network attack groups such as U2R and R2L category. The authors further proposed that NSL-KDD, a refined and condensed dataset of the original KDD'99 dataset constitutes the same 41 features and one class attribute which is composed of 21 classes which are covered under four classes of attacks such as, Probe, User to Root (U2R), Remote to Local (R2L) and Denial of Service (DoS). In this study, the multi-class NSL-KDD dataset is converted to the binary class dataset by combining different types of anomalies. So, now there is a binary class *i.e.*, normal and anomaly. The relative description of NSL-KDD dataset is explained in Figure 3.1.

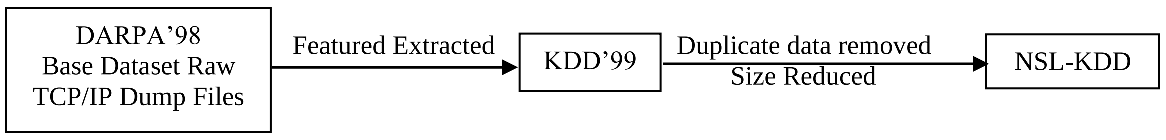


Fig.3.1- Relative Description of NSL-KDD dataset

There are 8 (Eight) different NSL-KDD dataset shown below in Table- 3.1 which are used for the research work for intrusion detection.

Table- 3.1: List of different NSL-KDD dataset with Description

Sl.No	Name of the file	Description of the file
1	KDDTrain+.ARFF	The full NSL-KDD train set with binary labels in ARFF format
2	KDDTrain+.TXT	The full NSL-KDD train set including attack-type labels and difficulty level in CSV format
3	KDDTrain+_20Percent.ARFF	A 20% subset of the KDDTrain+.arff file
4	KDDTrain+_20Percent.TXT	A 20% subset of the KDDTrain+.txt file
5	KDDTest+.ARFF	The full NSL-KDD test set with binary labels in ARFF format
6	KDDTest+.TXT	The full NSL-KDD test set including attack-type labels and difficulty level in CSV format
7	KDDTest-21.ARFF	A subset of the KDDTest+.arff file which does not include records with a difficulty level of 21 out of 21
8	KDDTest-21.TXT	A subset of the KDDTest+.txt file which does not include records with a difficulty level of 21 out of 21

Source: Dhanabal and Shantharajah, 2015.

3.2.2 Types of Attack in NSL-KDD Dataset

There are four types of attacks in the network domain. The attack types with corresponding attack name in NSL-KDD dataset is shown in Table-3.2.

Table-3.2 Attack types with corresponding attack name in NSL-KDD dataset

Type of Attack	**Attack Name**
Denial of Sevice (DoS)	Back, land, Neptune, pod, smurf, teardrop
Remote to Local (R2L)	guess_passwd, ftp_write, imap, phf, multihop, warezmaster
User to Root(U2R)	buffer_overflow, loadmodule, perl, rootkit
Probing	Satan, ipsweep, nmap, portsweep

3.3 Experimental Setup

The whole neural network experiments are performed using Knowledge Extraction based on Evolutionary Learning (KEEL), an open-source (GPLv3) Java software that is used for a large number of different knowledge data discovery tasks. The details of the experimental set-up have been described in Chapter-2, Section 2.16.2. To test different classification algorithms of Artificial Neural Network, the original NSL-KDD Dataset which is an enhanced edition of the original KDD'99 dataset is used for the whole experiments that consist of 41 features as well as 25190 training instances. All selected 6 different classifiers from various classification techniques of Artificial Neural Network were tested. In this study, a 10-fold cross-validation technique is applied, where 9 folds are used for training purposes and 1-fold is used for test purposes for comparison between various Artificial Neural Networks algorithms to know the best performance for Intrusion Detection System.

The various parameters of different algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN used for the experimental setup are discussed threadbare in Chapter-2, Table 2.6.

One common method of determining the performance of a classifier is performed through using a confusion matrix (Kohavi and Provost, 1998). The performance of the different classifiers is monitored and measured by using various types of performance matrices such as i) Overall Accuracy, ii) Specificity, iii) Sensitivity, iv) G-Mean (Kubat *et al.*, 1998), v) Precision, vi) Recall, vii) F-Measure (Lewis and Gale, 1994) and viii) Matthews's Correlation Coefficient (MCC). A brief description of various performance matrices as stated above used for the study has discoursed in Chapter-2 Section-2.17.

3.4 Mathematical Formulae for all Performance Matrices

The mathematical formulae for all the performance matrices used to measure the performance of the different classifiers are placed below in Table- 3.3.

Table 3.3: Mathematical Formulae for all the Performance Matrices

$Overall\ Accuracy\ (OV) = \frac{TP + TN}{TP + FP + FN + TN}$	(1)
$Specificity = \frac{TN}{TN+FP}$	(2)
$Sensitivity = \frac{TP}{TP + FN}$	(3)
$G - Mean = \sqrt{Specificity * Sensitivity}$	(4)
$Precision = \frac{TP}{TP + FP}$	(5)
$Recall = \frac{TP}{TP + FN}$	(6)
$F = 2.\frac{Precision \cdot Recall}{Precision + Recall}$	(7)
$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$	(8)

Sources: Fawcett (2006), Powers (2011), and Ting (2011)

Where,	TP = Total number of correctly classify positive examples
	FP = Total number of miss-classified negative examples
	TN = Total number of correctly classify negative examples
	FN = Total number of miss-classified positive examples

3.5 KEEL-Dataset Repository- An Introduction

KEEL (Knowledge Extraction based on Evolutionary Learning) 3.0, an open-source (GPLv3) Java software originally developed in 2009 was upgraded in 2009 as a tool for the implementation of evolutionary algorithms and soft computing techniques for standard Data Mining problems such as regression, classification, or association rules, as well as data pre-processing technique (Triguero *et al.*, 2017). The KEEL Suite 3.0 comprises five different blocks such as i) Data Management, ii) Experiments, iii) Educational, iv) Modules and v) Help. A detailed description of the different blocks is mentioned in Chapter-2 Section-2.18.

3.6 Proposed Artificial Neural Network Experimental Design

For the experiments of various neural network algorithms, an experimental framework has been designed in Fig.-3.3 below. The proposed framework consists of network traffic data, data pre-process, various neural network algorithms, and different performance matrices are used to measure the performance and the details are described in Table-3.3 above.

3.6.1 Data pre-process

The network traffic dataset was initially prepared, analysed, and converted from multiclass to binary by combining all the attack types and to make it normal and anomaly. After pre-processing and reducing redundant data, 25190 instances are selected for the experimentation dataset. For the whole experimentation, full 41 features were obtained from NSL-KDD dataset. The detailed description of the dataset used for training and testing experiments is dealt with in Table-3.4.

3.6.2 Experimental Framework

In this experiment, NSL-KDD dataset are taken as input for the different algorithms i.e., LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN. To evaluate the performance of different algorithms, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and Matthews's Correlation Coefficient are used. The architecture of the experimental framework is placed below in Figure-3.2.

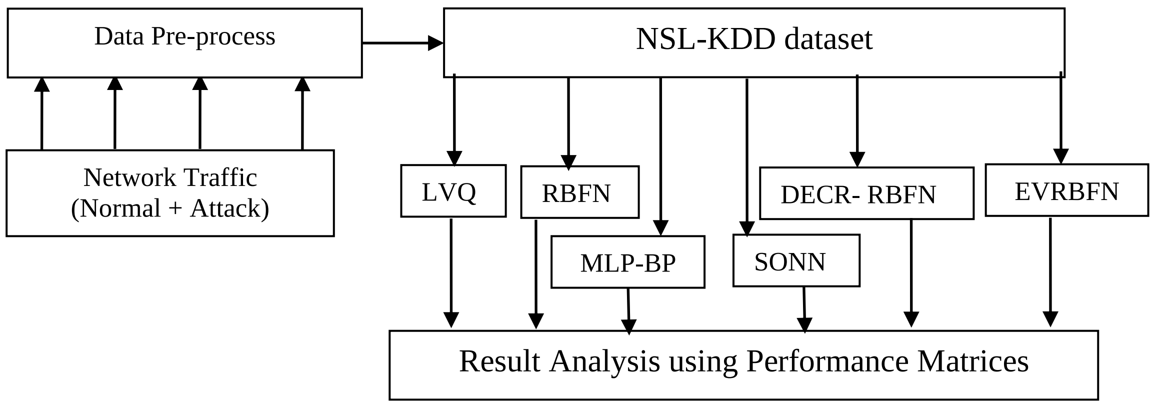

Data Pre-process
NSL-KDD dataset
Network Traffic (Normal + Attack)
LVQ
RBFN
DECR- RBFN
EVRBFN
MLP-BP
SONN
Result Analysis using Performance Matrices

Fig. 3.2: Experimental Framework of Artificial Neural Network

3.7 NSL-KDD Dataset for Artificial Neural Network Experiments (Training and Test)

The NSL-KDD (training and test) dataset are analyzed after pre-processing and reducing redundant data, 25190 input instances are selected for experimentation purposes. Then trainset was divided into ten sets randomly, containing both normal and attack data that appears in the NSL-KDD dataset.

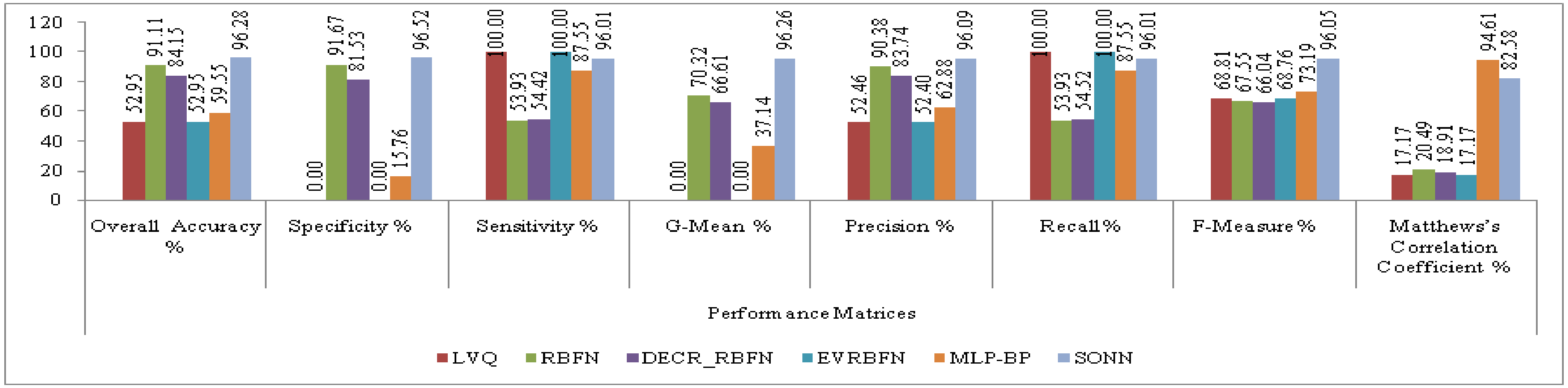
The different attacks contained in NSL-KDD dataset are back, land, Neptune, pod, smurf, teardrop, satan, ipsweep, nmap, portsweep, guess_password, ftp_write, imap, phf, multihop, warezmaster, warezclient, spy, buffer_overflow, load-module, snmpgetattack, xlock, sendmail, apache2, udpstorm, xsnoop, xterm, mscan, processtable, httptunnel, mailbomb, ps, snapgueuss, named, saint, perl, and rootkit, etc. These all-attack types fall under four attack categories i.e., Denial of Service (Dos), User to Root (U2R), Remote to Local (R2L), and Probe. Out of ten dataset, nine dataset were selected for training and one dataset was selected randomly for the test. The training dataset comprise total instances, 929511 (2519*41* 9=929511) while, the test dataset constitutes the total instances, 103279 (2519*41= 103279). The program itself randomly selects the data for training and testing purposes. The distribution of data for NSL-KDD dataset is shown in Table-3.4.

Table 3.4: Distribution of data for NSL-KDD (training & test) experiments

Dataset Name	No. of Features	Input instances	Total
Trainset data 1	41	2519	2519* 41*1= 103279
Trainset data 2	41	2519	2519* 41*1= 103279
Trainset data 3	41	2519	2519* 41*1= 103279
Trainset data 4	41	2519	2519* 41*1= 103279
Trainset data 5	41	2519	2519* 41*1= 103279
Trainset data 6	41	2519	2519* 41*1= 103279
Trainset data 7	41	2519	2519* 41*1= 103279
Trainset data 8	41	2519	2519* 41*1= 103279
Trainset data 9	41	2519	2519* 41*1= 103279
	Total Training instances		**929511**
Testset data 1	41	2519	2519 *41*1= 103279
	Total Test instances		**103279**

Table- 3.5: Experimental Results of Artificial Neural Network Algorithms using NSL- KDD (training) dataset

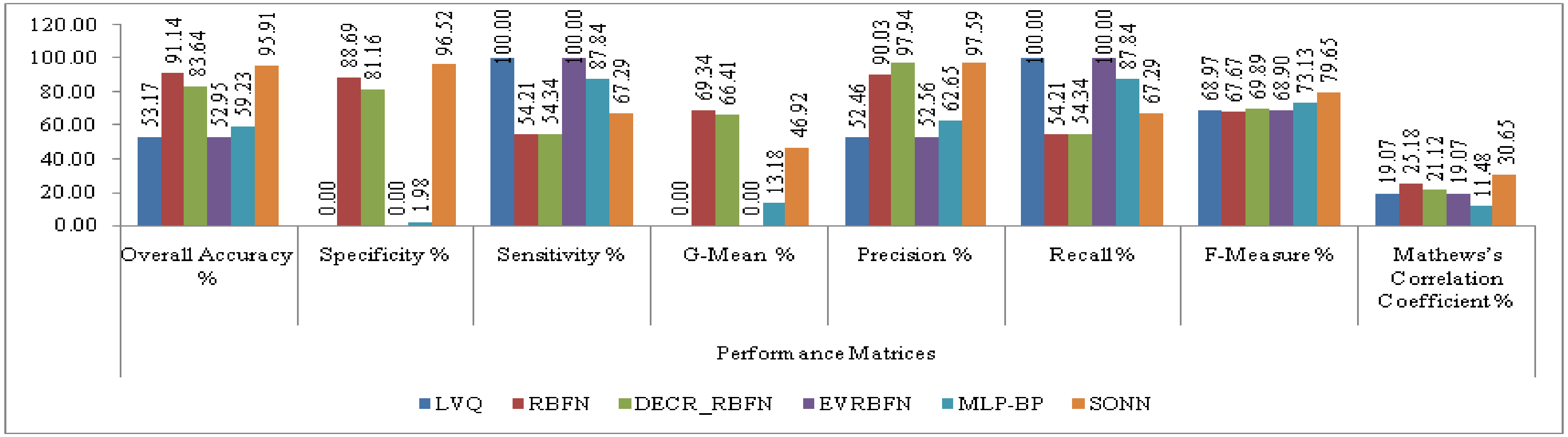
Algorithms	Performance Matrices							
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-Measure %	Matthews's Correlation Coefficient %
LVQ	52.95	00.00	100.00	00.00	52.46	100.00	68.81	17.17
RBFN	91.11	91.67	53.93	70.32	90.38	53.93	67.55	20.49
DECR-RBFN	84.15	81.53	54.42	66.61	83.74	54.52	66.04	18.91
EVRBFN	52.95	00.00	100.00	00.00	52.40	100.00	68.76	17.17
MLP-BP	59.55	15.76	87.55	37.14	62.88	87.55	73.19	94.61
SONN	96.28	96.52	96.01	96.26	96.09	96.01	96.05	82.58



Graph 3.1: Experimental Results of Artificial Neural Network Algorithms using NSL- KDD (training) dataset

Table-3.6: Experimental Results Artificial Neural Network using NSL- KDD (test) dataset

Algorithms	Performance Matrices							
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-Measure %	Matthews's Correlation Coefficient %
LVQ	53.17	00.00	100.00	00.00	52.46	100.00	68.97	19.07
RBFN	91.14	88.69	54.21	69.34	90.03	54.21	67.67	25.18
DECR-RBFN	83.64	81.16	54.34	66.41	97.94	54.34	69.89	21.12
EVRBFN	52.95	00.00	100.00	00.00	52.56	100.00	68.90	19.07
MLP-BP	59.23	01.98	87.84	13.18	62.65	87.84	73.13	11.48
SONN	95.91	96.52	67.29	46.92	97.59	67.29	79.65	30.65



Graph-3.2: Experimental Results Artificial Neural Network using NSL- KDD (test) dataset

3.8 Experimental Results

In this section**,** the experimental results of six different Artificial Neural Network algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN are discussed. To evaluate the performance of these algorithms' different performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and Matthews's Correlation Coefficient are used. The evaluated results for both training and test were obtained after proper validation of the experiments.

3.8.1 Experimental Results of NSL-KDD (training) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews's Correlation Coefficient for six different neural network algorithms i.e, LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN using NSL-KDD (training) dataset as stated in section 3.7 is placed in Table- 3.5 with the corresponding Graph- 3.1.

Table- 3.5 shows that the SONN algorithm has performed the best result for Overall Accuracy i.e., 96.28%. It means that this algorithm accurately classifies the training dataset as compared to other algorithms. But the algorithm, RBFN classifies the instance with 91.11%, whereas other algorithms such as DECR-RBFN and MLP-BP algorithms resulted in 84.15% and 59.55% respectively. It means the MLP-BP algorithm performs very low as compared with others. Again, LVQ and EVRBFN show very low performance i.e. 59.95% each while classifying the training dataset.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e. 96.52% which otherwise means that the SONN algorithm correctly classifies both positive and negative classes. But other algorithms like RBFN classify the dataset with 91.67% whereas, DECR-RBFN with 81.53%. But, the MLP-BP algorithm classifies the training instance with 15.76%, which also reveals very low performance as compared to other algorithms it means that this algorithm could not classify the dataset accurately, whereas, both LVQ and EVRBFN algorithm comes with 0.0%.

The result for Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm shows the best performance i.e. 100% each. It means that both the LVQ and EVRBFN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithm like SONN shows 96.01% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like MLP-BP, DECR-RBFN, RBFN performed 87.55%, 54.42%, and 53.93% results respectively, which otherwise means that RBFN algorithm shows very low performance as it could not classify the training dataset correctly.

For the result of G-mean or G-measure, the SONN algorithm performed the best result i.e. 96.26% among others, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (96.52%) and Sensitivity (96.01%) of the SONN algorithm are high. So, the G-mean of SONN (96.26%) is also reasonably high. Further, the G-mean value for RBFN constitutes 70.32% because the value of both Specificity (91.67%) and Sensitivity (53.93%) is also higher. But the performance of an algorithm like DECR-RBFN is 66.61%, which is not very good because of its Sensitivity value i.e. 54.42%. The performance of the MLP-BP algorithm is 37.14%, which is not very good because of its Specificity value i.e. 15.76%. Finally, the performance of LVQ and EVRBFN comes with 0.0% each because the Specificity value of each algorithm is also 0.0%.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 96.09% of exactness or quality of the classifier. But the other algorithms like RBFN (90.38%) of exactness, whereas DECR-RBFN (83.74%), MLP-BP (62.88%), LVQ (52.46%), EVRBFN (52.40%) of exactness. Ultimately it deduced that the MLP-BP algorithm shows very low performance as compared with other algorithms.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, the result of Recall or Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm shows the best performance i.e. (100%) each. It means that both the LVQ and EVRBFN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithm like SONN shows 96.01% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like MLP-BP, DECR-RBFN, RBFN performed 87.55%, 54.42%, and 53.93% results respectively, which otherwise means that RBFN algorithm shows very low performance as it could not classify the training dataset correctly.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here the SONN algorithm gives the best result i.e. 96.05% because the result for both Precision and Recall is relatively very high. i.e. 96.09% and 96.01% respectively. Then the result for the MLP-BP algorithm shows 73.19% because the result for both Precision and Recall also shows very good. i.e. 62.88% and 87.55%. Then the result of LVQ and EVRBFN algorithm comes to 68.81% and 68.76% respectively. But, for RBFN and DECR-RBFN algorithms, the result of F-measure comes to 67.55% and 66.04% respectively only because of their corresponding Recall value, which is very low i.e. 53.93% and 54.52%.

The result of MCC for MLP-BP algorithm shows the best result among all other algorithms i.e. 94.61% while, the SONN algorithm also shows a very good result for MCC i.e. (82.58%). But the other algorithms like RBFN, DECR-RBFN, LVQ, EVRBFN algorithms show very low performance i.e. 20.49%, 18.91%, 17.17%, and 17.17% respectively.

3.8.2 Experimental Results of NSL-KDD (test) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews’s Correlation Coefficient for six different neural network algorithms i.e, LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN using NSL-KDD (test) dataset as stated in section 3.7 is placed in Table- 3.6 with the corresponding Graph-3.2.

The table shows that the SONN algorithm has performed the best result for Overall Accuracy i.e. 95.91%. It means that this algorithm accurately classifies the test dataset as compared to other algorithms. But the algorithm, RBFN classifies the instance with 91.14%, whereas other algorithms such as DECR-RBFN and MLP-BP algorithms resulted in 83.64% and 59.23% respectively. Then the other algorithm like LVQ and EVRBFN the result of Overall Accuracy comes to 53.17% and 52.95% correspondingly. It means the EVRBFN algorithm performs very low as compared with others while classifying the test dataset.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e. 96.52% which otherwise means that the SONN algorithm correctly classifies both positive and negative classes. But other algorithms like RBFN classify the dataset with 88.69% whereas, DECR-RBFN with 81.16%. But, the MLP-BP algorithm classifies the test instance with 01.98%, which also reveals very low performance as compared to other algorithms it means that this algorithm could not classify the dataset accurately, whereas, both LVQ and EVRBFN algorithm comes with 0.0%.

The result for Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm shows the best performance i.e. 100% each. It means that both the LVQ and EVRBFN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithm like MLP-BP shows 87.84% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like SONN, DECR-RBFN, RBFN performed 67.29%, 54.34%, and 54.21% results respectively, which otherwise means that RBFN algorithm shows very low performance as it could not classify the test dataset correctly.

For the result of G-mean or G-measure, the RBFN algorithm performed the best result i.e. 69.34% reasonably high among others, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (88.69%) and Sensitivity (54.21%) of the RBFN algorithm are high. Further, the G-mean value for DECR-RBFN constitutes 66.41% because the value of both Specificity (81.16%) and Sensitivity (54.34%) is also higher. But the performance of an algorithm like SONN is 46.92%, which is not very good because of its Sensitivity value i.e. 67.29%, even if, its Specificity value is so high i.e.96.52%. The performance of the MLP-BP algorithm is 13.18%, which is very low performance because of its Specificity value i.e. 01.98%. Finally, the performance of LVQ and EVRBFN comes to 00% each because the Specificity value of each algorithm is also 0.0%.

The result of Precision or Positive Predictive Value, DECR-RBFN algorithm reveals the best result among all other algorithms i.e. 97.94% of exactness or quality of the classifier. But the other algorithms like SONN (97.59%) of exactness, whereas RBFN (90.03%), MLP-BP (62.65%), EVRBFN (52.56%), LVQ (52.46%) of exactness. Ultimately it deduced that the LVQ algorithm shows very low performance as compared with other algorithms.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, the result of Recall or Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm shows the best performance i.e. 100% each. It means that both the LVQ and EVRBFN algorithm accurately classifies the true positive instances or positive classes of the present dataset whereas, the other algorithm like MLP-BP shows 87.84% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like SONN, DECR-RBFN, RBFN performed 67.29%, 54.34%, and 54.21% results respectively, which otherwise means that RBFN algorithm shows very low performance as it could not classify the test dataset correctly.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here the SONN algorithm gives the best result i.e. 79.65% because the result for both Precision and Recall is relatively very high. i.e. 97.59% and 67.29% respectively. Then the result for the MLP-BP algorithm shows 73.13% because the result for both Precision and Recall also shows very good. i.e. 62.65% and 87.84% respectively and the result of DECR-RBFN and LVQ algorithm comes to 69.89% and 68.97% respectively. This result is not considered as so good performance because the Recall value of DECR-RBFN is 54.34% and the Precision value of LVQ is 52.46%. But, for EVRBFN and RBFN algorithms, the result of F-measure comes to 68.90% and 67.67% respectively only because of the Precision value of EVRBFN and Recall value of RBFN, which are very low i.e. 52.56% and 54.21% respectively.

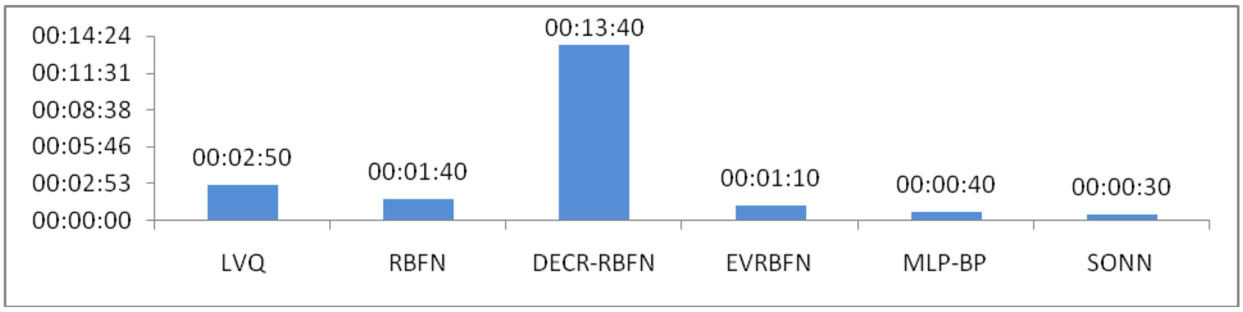
The result of MCC for the SONN algorithm shows, the best result among all other algorithms which comes to 30.65%. Surprisingly, the RBFN algorithm also shows very good performance for MCC i.e. 25.18%. But the other algorithms like DECR-RBFN, LVQ, EVRBFN, and MLP-BP show very low performance i.e. 21.12%, 19.17%, 19.07%, and 11.48% respectively.

3.8.3 Training time for Experiment

The training time consumed for the experiments of the different Artificial Neural Network algorithms to train the NSL-KDD dataset is shown in Table-3.7 with the corresponding Graph-3.3. From this table, we deduced that the SONN algorithm is taking the optimal time i.e., 00:00:30 to train the dataset.

Table-3.7: Training time of the Artificial Neural Network Algorithms

Algorithms	Training Time
LVQ	00:02:50
RBFN	00:01:40
DECR-RBFN	00:13:40
EVRBFN	00:01:10
MLP-BP	00:00:40
SONN	00:00:30



Graph- 3.3: Training time of the Artificial Neural Network Algorithms

3.9 Conclusion

The performance of various neural network classification algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN have been evaluated based on the NSL-KDD dataset with full 41-dimension features was used throughout the experiments and compared. To measure the performance of the different algorithms, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews's Correlation Coefficient (MCC) were used.

It was found from the experimental results that the algorithm performs well on the original KDD'99 dataset. But it does not produce the same result for the NSL-KDD dataset. This, otherwise proves that the NSL-KDD dataset represents the more accurate result for the evaluation of different neural network algorithms. Among various tested classification algorithms, the SONN algorithm shows the best result (Overall Accuracy) compared to other tested algorithms. According to the time constraints again SONN algorithm shows the best result to train the dataset.

CHAPTER-4

EXPERIMENTS USING SUPPORT VECTOR MACHINE (SVM)

Chapter-4

Experiments Using Support Vector Machine (SVM)

4.1 Introduction

Intrusion detection is also equally an important research issue in network security that signifies the process of monitoring and also analyzing network traffic data to observe security infringements. Both mining and machine learning approaches are globally acceptable podiums to extend an intrusion detection system as it leads to inspect network traffics and aware the administrator in case of unauthorized access or attempts by an outsider. The network traffic can be classified into normal and anomalous to detect intrusions.

4.2 Algorithms

In this study, some linear and kernel classification algorithms from the family of Support Vector Machine have been exercised for the experiments and performance evaluation. Based on their evaluation performance, the top 6 (six) best set of supervised linear and non-linear Support Vector Machine classifiers are selected and employed. Some linear and kernel classification techniques consisting of Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, Coarse Gaussian SVM algorithms. All the above classification techniques are evaluated and compared the performance using both KDD and NSL-KDD dataset. The detailed theoretical descriptions of all support vector machine classifiers are described below.

4.2.1 Support Vector Machine (SVM)

The Support Vector Machine model which was first introduced by Boser *et al.* (1992), is a state-of-the-art pattern recognition technique and it has numerous application areas such as, optical character recognition, object detection, face verification, text categorization, etc. (Sivanandam *et al.*, 2006). Support Vector Machine is an innovative approach to constructing a learning mechanism that minimizes the generalization error and is based on simple and intuitive concepts.

The dominance of the Support Vector Machine which was used as a pattern classifier of statistical learning technique for both classification and the regression problems expanded its horizon effectively to several pattern recognition applications using a variety of kernel functions. It is recognized as a viable and effective instrument for intrusion detection in the information security domain. The significance of the Support Vector Machine in the present information security context is widely recognized due to its general applications and the ability to detect problems. Further, the use of the Support Vector Machine is more significant for determining the actual risk even in high-dimensional spaces with small samples using kernel tricks. It also appropriately selects the different setup parameters as it abstains from risk as it is found in neural networks (Jha and Ragha, 2013).

An early machine learning algorithm is aimed at learning representations of simple functions. Hence, the goal of learning was to output a hypothesis that performed the correct classification of the training data and early learning algorithms were designed to find such an accurate fit to the data. The indispensable idea behind Support Vector Machine is that the input vectors are mapped in a non-linear method into a high dimensional feature space and it determining the best possible hyperplane, which separates the data into classes such as that the “margin” or distance between the members of different classes is maximized. (Cristianini and Taylor, 2000). The ability of a hypothesis to correctly classify data, not in the training set is known as its generalization. Support Vector Machine performs better in terms of not overgeneralization when the neural networks might end up over-generalizing easily (Mitchell, 1997).

Given a set of binary data that contains the examples from two classes denoted as {+, -} which are linearly separable and from this illustration one can easily define a linear separating boundary or hyperplane as shown in Figure-4.1.

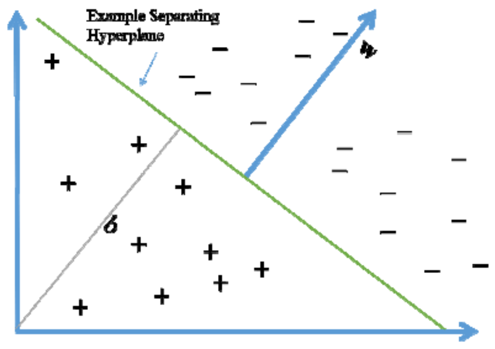


Fig.4.1-Example of a Separating hyperplane for a two-dimensional dataset
Source: Cristianini and Taylor, 2000

In this case, the set of numbers belongs to two classes, either the "+" class or the "-" class which can also be considered as the "+1" class or the "-1" class correspondingly. A "decision surface" is defined such that a "hyperplane" will correctly distinguish the input samples into correct classes by the following equation

1. $f(x_i) = w^T x + b$ (1)

(Where for $f(x_i) \geq 0$ the sample belongs to the + class and for $f(x_i) < 0$ to the sample belongs the − class)

Note that, there are infinitely several separating lines or hyperplanes that would successfully divide the "+" class inputs from the "-" class inputs. But Support Vector Machine always establishes the "optimal" technique to determine the separating hyperplane.

The basic learning approach utilized by Support Vector Machine is primarily based on an optimal learning algorithm, which is derived from statistical learning theory and has a quadratic programming solution. The Support Vector Machine is an influential machine learning procedure, which was earlier developed only for the solution towards the binary classification problems, and later it was extended to solve the regression problems. Based on statistical learning theory, Support Vector Machine map input vectors in a non-linear manner into a high dimensional feature space such that the best possible separating hyperplane can be found (Cortes and Vapnik, 1995).

Support Vector Machine is based on the idea of "maximizing the margin" among the classes in a set of training data. This idea was initially offered by Vapnik(1992) from Bell Laboratories. The author also introduced to design the idea of supporting patterns and suggested the mathematics for the training algorithm. (Boser *et al.*, 1992). The optimal hyperplane and Support Vectors for Linearly Separable data are represented below in Fig.4.2.

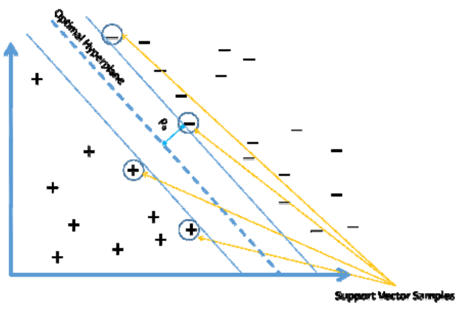


Fig. 4.2-Optimal hyperplane and Support Vectors for Linearly Separable Patterns
Source: Haykin, 1999

Linear Support Vector Machine is the recent origin and it is a fast machine learning (data mining) algorithm to solve multiclass classification problems from ultra-large dataset that implement an original proprietary version of a cutting plane algorithm for designing a linear support vector machine (Huang and Kecman, 2009).

The following equations describe the decision surface, where ***x*** is an input vector, and ***w*** is a variable weight vector and *b* is a bias value.

$$w^T x + b = 0 \qquad (1)$$
$$w^T x + b \geq 0 \text{ for} \qquad d_i = +1 \qquad (2)$$
$$w^T x + b < 0 \text{ for} \qquad d_i = -1 \qquad (3)$$

The principal purpose of a Support Vector Machine is to find an optimal hyperplane so that, the margin from the nearest component of the class {-1, +1} is maximized as derived in figure-2.9. Hence, the optimal hyperplane implies that the distance between the hyperplane the adjacent data point is (i.e. “margin of separation”, ρ_0) is maximized. (Haykin, 1999).

4.2.1.1 Kernel Trick

If the given input dataset is linear, a separating hyperplane may be used to distinguish the input data. However, it is often the case that the given dataset is far from linear and the dataset are inseparable. To allow for these kernels are used to non-linearly map the input data to a high-dimensional space. The new mapping is then linearly separable. A diagram of this is shown below in Fig.4.3 (Mitchell, 1997).

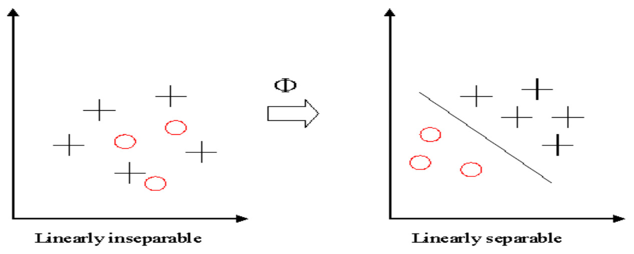


Fig. 4.3-Diagram of linear and nonlinear classification

This mapping function is defined by the Kernel. $k(x,y) = \emptyset(x).\emptyset(y)$. Feature Spaceindicates the transformation of the input data into feature space in such a way that, it makes it possible to define a similarity measure based on the dot product. If the feature space is selected correctly, then the pattern recognition task is very easy (Mitchell, 1997). The Kernel trick usually allows the Support Vector Machine to form nonlinear boundaries for the input samples. Then the kernel represents a valid inner product in the feature space. The training set is not linearly separable in the input space. Then the training set is linearly separable in the feature space. This is called the “Kernel Trick”. The steps involved in the kernel trick are discussed below (Burges, 1998).

(a) The algorithm is always expressed by using only the inner products of data sets. This is also called a dual problem.

(b) Original dataset is passed through nonlinear maps. To form new data concerning new dimensions, by adding a pairwise product of some of the original data dimensions to each data vector.

(c) Rather than an inner product on these new, larger vectors and stored in tables and later do a table lookup. Then it can represent a dot product of the data after doing nonlinear mapping on them. This function is the kernel function.

4.2.1.2 Kernel Trick and Functions: Inner Product summarization

Even if the dot product of nonlinearly mapped data can be expensive, but to represent the dot product of the data vectors, it is very much required. But the kernel trick just picks a suitable function that corresponds to the dot product of some nonlinear mapping. A particular kernel technique is only chosen by the trial-and-error method only on the test dataset. To choose the right kernel technique based on the problem or application which would help to enhance Support Vector Machine's performance (Burges, 1998).

The various kernel functions are described below. The below-mentioned functions which are the most commonly used kernel types were extracted from the various explanation on kernel functions. (Cristianini and Taylor, 2000 & Hsu *et al.*, 2003).

(i) Linear Kernel: $k(x_i,x_j)= x_i^T . x_j$, which is the special case of the Polynomial Kernel function. Where $\gamma=1$, $r=0$, $d=1$.

(ii) Polynomial Kernel: $k(x_i, x_j)=(\gamma x_i^T . x_j + r)^d, \gamma > 0$

(iii) Gaussian/Radial Basis Function (RBF) Kernel:
$k(x_i, x_j)= e(\gamma||x_i - x_j||)^2 \ \gamma > 0$

For main implementations of Polynomial Kernel functions, the exponent d is an integer that is always greater than one. The thought of using the kernel functions with d is less than one, which is usually known as a fractional polynomial. It is primarily used for improving the performance of the Support Vector Machine. The Gaussian kernels are always non-linearly mapping the data space into a higher dimensional space (Rossius, 1998).

The kernel matrix $k(x_i, x_j)$ provides an "implicit mapping" of the input samples to an inner product form of the "feature space". This is called the "Kernel Trick", which allows avoiding computation in the high dimensional feature space as the kernel is computed from inner products only. (Cristianini and Taylor, 2000). It is always possible to distinguish the input data into separate classes with a particular hyperplane, by mapping the training data from the original input space into a higher-dimensional space. The key idea behind this is, the original feature space can always be mapped to several higher-dimensional feature spaces where the training dataset is linearly separable. The mapping function from feature space to a high dimensional space is given in Fig-4.4.

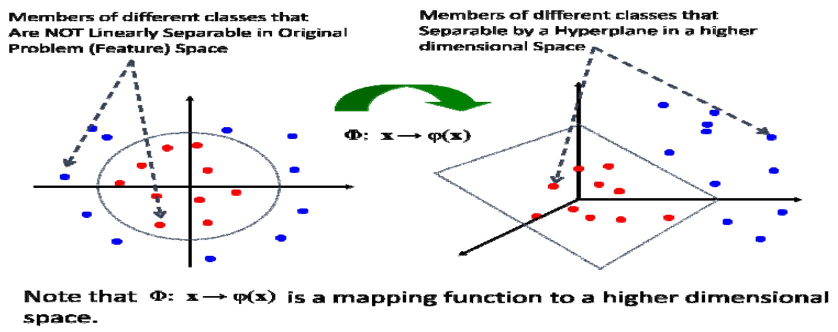

Members of different classes that Are NOT Linearly Separable in Original Problem (Feature) Space
Members of different classes that Separable by a Hyperplane in a higher dimensional Space
Φ: x ⟶ φ(x)
Note that Φ: x ⟶ φ(x) is a mapping function to a higher dimensional space.

Fig.4.4 Mapping function of non-separable data into higher dimensional space Source-Eck, 2006

Usually, kernels must satisfy these two following conditions. (Bartlett, 2008).

(i) k is always symmetric means, $k(x_i, x_j)= k(x_i, x_j)$

(ii) For all $x_1 \ldots\ldots x_N$ in the training dataset, $k(x_i, x_j)$ is always positive semi-definite. (Mercer, 1909)

A new quadratic kernel-free non-linear Support Vector Machine (which is called QSVM) function is capable of separating non-linearly the data is used. The geometrical margin is proved to be equal to the inverse of the norm of the gradient of the decision function. The functional margin is the equation of the quadratic function. QSVM is proved to be put in a quadratic optimization setting. This setting does not require the use of a dual form or the use of the Kernel trick (Dagher, 2008). In this study, another kernel method like Cubic SVM and some Gaussian kernel methods i.e. Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM are used.

The results for different classifiers corresponding to their memory usage, interpretability, and model flexibility are shown below in Table-4.1

Table-4.1: Information Description of Different Classifiers

Classifier Type	Prediction Speed	Memory Usage	Interpretability	Model Flexibility
Linear SVM	Binary: Fast Multiclass: Medium	Medium	Easy	Low Makes a simple linear separation between classes.

Quadratic SVM	Binary: Fast Multiclass: Slow	Binary: Medium Multiclass: Large	Hard	Medium
Cubic SVM	Binary: Fast Multiclass: Slow	Binary: Medium Multiclass: Large	Hard	Medium
Fine Gaussian SVM	Binary: Fast Multiclass: Slow	Binary: Medium Multiclass: Large	Hard	High Decreases with kernel scale setting. Makes finely detailed distinctions between classes, with kernel scale set to sqrt(P)/4.
Medium Gaussian SVM	Binary: Fast Multiclass: Slow	Binary: Medium Multiclass: Large	Hard	Medium Medium distinctions, with kernel scale set to sqrt(P).
Coarse Gaussian SVM	Binary: Fast Multiclass: Slow	Binary: Medium Multiclass: Large	Hard	Low Makes coarse distinctions between classes, with kernel scale set to sqrt(P)*4, where P is the number of predictors.

4.2.2 Application of SVM for Classification

The SVM is known as the most excellent learning algorithm for the binary classification method. For data classification, Support Vector Machine has proved to be pragmatic and influential even though, sometimes it gives a disappointing result compared to Neural Networks which are user-friendly. Invariably, taking the training and testing data that contain data instances, classification is performed and each instance in the training set comprises one target value and many attributes. Practically, Support Vector Machine is intended to develop a model that predicts the target value of data instances in the set of training data where attributes are available (Duda and Hart, 1973).

Further, classification in Support Vector Machine is a case of Supervised Learning. The recognized identity visualizes the proper functioning of the framework. The data focus on a coveted reaction by approving the precision of the framework or to be utilized for enabling the system to learn to act correctly. Support Vector Machine also includes in a stage the recognizable proofs that are associated with the known classes and this phenomenon is known as feature selection or feature extraction. Both feature selection and SVM classification have used also in the absence of the prediction of unknown samples. They can be used to distinguish the key sets that are associated with any process to distinguish the classes.

4.2.3 Application of SVM for Regression

Support Vector Machine can also be applied to regression problems. For this introduction of an alternative loss, a function is required. Then the loss function must be modified to comprise a distance measure. The regression problem can be linear and nonlinear in a manner. Linear models primarily consist of various loss functions. It includes (i) e-intensive loss functions, (ii) quadratic, and (iii) Huber loss function. (Smola, 1996). Similarly, to classification problems, a non-linear model is typically required to adequately model data. In the same manner, for the classification problem, the two approaches are usually used. It includes (i) non-linear SVC approach and (ii) non-linear mapping. These approaches can be primarily used to map the data into a high-dimensional feature space, where linear regression is performed. Again, the kernel approach is also employed to address the curse of dimensionality. In the regression method, many considerations are usually based on the earlier knowledge of that concerned problem and the distribution of the noise. In the absence of such information, Huber’s robust loss function has been proved to be a superior alternative (Cortes and Vapnik, 1995).

In this study, we limit our application of Support Vector Machine only for the classification problem. Support Vector Machine comprises many classification techniques out of which, for the present work six types of linear and nonlinear classifiers such as i) Linear SVM, ii) Quadratic SVM, iii) Cubic SVM, iv) Fine Gaussian SVM, v) Medium Gaussian SVM, and vi) Coarse Gaussian SVM has been used for the experiments based on their performances and accuracy for the Intrusion Detection System database.

4.3 Dataset used for Experiments

In this study, two types of dataset prepared from KDD’99 i.e, KDD Cup 1999 and NSL-KDD (Tavallaee *et al.*, 2009) Intrusion Dataset are used to evaluate each classification technique of Support Vector Machine.

4.3.1 KDD’99 Cup Dataset used in SVM

As already discussed, the KDD’99 cup dataset that comprises around 4,900,000 single connection vectors having every 41 features constitute and labeled as normal or an attack with one specific attack type. This dataset is used as a benchmark dataset in the field of network intrusion detection system studies (Stolfo *et al.*,2000 & Lippmann *et al.*, 2000). The details of the dataset distribution are placed KDD’99 is demonstrated in Chapter-1 Figure 1.4.

4.3.2 NSL-KDD Dataset used in SVM

As already discussed in Chapter-1 supplemented with detailed dataset description (both training and test) in Table-1.10, the NSL-KDD dataset, an improved without redundant records of KDD’99 Cup dataset is a refined and condensed dataset of original KDD’99 dataset constitutes same 41 features and one class attribute which is composed of 21 classes which are covered under four classes of attacks (Tavallaee *et al.*, 2009). In this study, the multi-class NSL-KDD dataset is converted to the binary class dataset by combining different types of anomalies. So, now, there is a binary class i.e., normal and anomaly.

4.4 Experimental Setup

The whole Support Vector Machine experiments are performed using Matrix Laboratory (MATLAB) Version 8.50.197613 (R2015a) with 64-bit (win 64) and Lic no. 161052 (Bryant and Garber, 1999; Sumathi and Paneerselvam, 2010) open-source software. We used Windows 7 Home Basic (64-bit) as the testbed operating system, Intel(R) Core (TM) i3 CPU M 370 @ 2.40GHz processor, 3GB of RAM. To test different classification algorithms of Support Vector Machine, both original KDD’99 and NSL-KDD Dataset, which is an enhanced edition of the original KDD’99 dataset, are used for the whole experiments consisting of 41 features. All selected 6 different classifiers from various classification techniques of SVM were tested based on the k-fold cross-validation (K-FCV) technique with each dataset. K-FCV is one of the most common methods, where the dataset gets divided into k, (where k represents the no. of folds or subsets), k-1 subsets are used as training sets, and k-(k-1) subset is used for the testing set. In this study, the 10-fold cross-validation technique is applied for comparison between various linear and nonlinear classifiers of Support Vector Machine to know the best performance for the Intrusion Detection System.

The performance of the classifiers are monitored and measured by using different performance matrices such as i) Overall Accuracy, ii) Specificity, iii) Sensitivity, iv) G-Mean, v) Precision (PPV), vi) Recall, vii) F-Measure, viii) Matthews’s Correlation Coefficient (MCC) and ix) ROC Curve. The performance of a classification algorithm is measured by Receiver Operating Characteristic (ROC Curve) or Area under Curve (AUC). The ROC graph is the trade-off between benefits and costs. A brief description of various performance matrices used for the study is mentioned below.

- **Overall Accuracy**

The overall accuracy of a classifier indicates how well the classifier classifies the training instance. This means that the classifier correctly classifies the positive and negative classes without any misclassification.

- **Specificity or TNR**

Specificity is also known as True Negative Rate (TNR) which means that the proportions of negative classes in a binary classification test are correctly identified. It is also known as ‘Inverse Recall’ and it is the proportions of real negative cases that are correctly predicted negative (Powers, 2007).

- **Sensitivity or TPR**

The sensitivity indicates a True Positive Rate (TPR) which means that the proportion of positive classes in a binary classification test is correctly identified. It is the proportion of Real Positives cases that are accurately predicted positive. This measures the coverage of the Real Positives cases by the +p (predicted positive) rule.

- **G-Mean**

The Geometric Mean (G-Mean) also known as G-measure is a metric that is used to evaluate the performance results by using both specificity and sensitivity. It ranges from 0 to 1 and an attribute that is perfectly correlated to the class provides a value of 1.

- **Precision or PPV**

Precision is also called a Positive Prediction Value (PPV) and it denotes the percentage of relevant objects that are identified for retrieval. The precision of a class is the number of true positives. It also denotes the amount of Predicted Positive cases or accurately Real Positives and is associated with the +P row.

- **Recall**

A recall is defined as the number of true positives divided by the total number of elements that truly belong to the positive class. The recall is also known as sensitivity is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Recall associates only to the +R column.

- **F-Measure**

The F-measure is also known as F1 score or F-score is a measure of a test's accuracy and is defined as the weighted harmonic mean or average of both precision and recall of the test, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. A high F-Measure value ensures that both recall and precision are reasonably high. F-measure successfully references the True Positives to the Arithmetic Mean of Predicted Positives and Real Positives (Ingre and Yadav, 2015).

- **Matthews's Correlation Coefficient (MCC)**

Matthews's correlation coefficient (Matthews, 1975) is employed in machine learning as a measure of the quality of binary (two-class) classifications. The MCC is, in essence, a correlation coefficient between the observed and predicted binary classifications, which returns a value between −1 and +1. A coefficient of +1 represents an ideal prediction, 0 no improved than random prediction and −1 indicates the whole difference between prediction and observation.

- **ROC Curve and AUC**

Receiver Operating Characteristic Curve (ROC) and its Area under the Curve (AUC) is a graphical plot, which illustrates the diagnostic capability of a binary classifier system, which is intended for using the performance analysis of the classifier. The ROC graph is represented through a plot between the false positive rate on '*X*' axis and the true positive rate on '*Y*' axis or it is the trade-off between benefits and costs. The point (0, 1) in the ROC Curve corresponds to the perfect classifier. This means that all the positive cases and negative cases are correctly classified. In point (0, 1) '0' stands for the false positive rate i.e. none while, '1' represents the true positive rate i.e, all. Likewise, in the point (0,0) '0' represents a classifier that predicts all cases to be negative and point (1,1) predicts to a classifier that, every case is positive. But the point (1,0) corresponds to a classifier incorrect for all classifications. The ROC Curve is also recognized as a relative operating characteristic curve because it is a comparison between two important operating characteristics i.e. TPR and FPR or the ROC graph is the trade-off between benefits and costs (Fawcett, 2006).

4.4.1 Mathematical formulae for all the performance matrices used to measure the performance of the different classifiers

The mathematical formulae for all the performance matrices used to measure the performance of the different classifiers are placed below in Table- 4.2.

Table 4.2: Mathematical Formulae for all the Performance Matrices

$OverallAccuracy\ (OV) = \frac{TP + TN}{TP + FP + FN + TN}$	(1)
$Specificity = \frac{TN}{TN + FP}$	(2)
$Sensitivity = \frac{TP}{TP + FN}$	(3)
$G - Mean = \sqrt{Specificity * Sensitivity}$	(4)
$Precision = \frac{TP}{TP + FP}$	(5)
$Recall = \frac{TP}{TP + FN}$	(6)
$F = 2.\frac{Precision \cdot Recall}{Precision + Recall}$	(7)
$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$	(8)

Sources: Fawcett (2006), Powers (2011) & Ting (2011)

Where	TP = Total number of correctly classify positive examples
	FP = Total number of miss-classified negative examples
	TN = Total number of correctly classify negative examples
	FN = Total number of miss-classified positive examples

One common method of determining the performance of a classifier is performed through using a confusion matrix (Kohavi and Provost 1998). When the performances of both classes are exercised, both True Positive Rate (sensitivity) and True Negative Rate (specificity) are high along with G-Mean or G-measure value. The value measures the balanced performance of a learning classifier between these two classes. The True Positive Rate (TPR) is also defined as Recall (R), which denotes the percentage of retrieved objects that are relevant, and Positive Predictive Value is defined as Precision (P), which denotes the percentage of relevant objects that are identified for retrieval. A high F-Measure value ensures that both Recall and Precision are reasonably high (Ingre and Yadav, 2015).

4.5 MATLAB- An Introduction

MATLAB, the acronym for Matrix Laboratory is a powerful and widely used open-source software to solve various types of problems in Mathematics, Computer Science, and Engineering, etc. It constitutes a wide range of predefined programs and functions which facilitate the engineers and scientists to deduce faultless results in the research conveniently. It constitutes a wide range of toolboxes for carrying out various functions and the toolboxes include Bioinformatics, Code Generation from MATLAB, Communication System, Computer Vision System, Control System, Curve Fitting, and many more. The different toolbox contains different functions. Programming in MATLAB has several advantages over other computer languages. A new program can be developed easily using the predefined functions, which in turn makes it much shorter in length. MATLAB usually manages memory automatically and it is especially practical for managing large data processing. MATLAB program can be run invariably on all platforms. But being an interpreted language, it runs slowly if it is not optimized properly through factorization while executing loop statements. It supports several windows to help a user to develop a program easily. MATLAB is generally used for (http://www.merl.com/papers/docs/TR94-03.pdf),

i. Mathematics and Calculation;
ii. Development of Algorithms;
iii. Prototyping, Modelling, and Simulation;
iv. Data Analysis, Exploration, and Visualization;
v. Scientific and Engineering Graphics;
vi. Application/Software Development.

4.6 Proposed Support Vector Machine Experimental Design

For the experiments of various Support Vector Machine classifiers, an experimental framework has been designed in Fig. 4.5. The proposed framework consists of network traffic data, data pre-process, various Support Vector Machine classifiers, and different performance matrices are used to measure the performance and the details are described in section 4.6.2.

4.6.1 Data pre-process

The network traffic dataset was initially prepared, analyzed, and converted from multiclass to binary by combining all the attack types and to make it normal and anomaly. After pre-processing and reducing redundant data, 25190 instances are selected for the experimentation dataset. For the whole experimentation, full 41 features were obtained from both KDD and NSL-KDD dataset. The detailed description of the dataset used for training and testing experiments is dealt with in section 4.7.

4.6.2. Experimental Framework

In this experiment, both KDD and NSL-KDD dataset are taken as input for the different classifiers i.e., Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, Coarse Gaussian SVM. To evaluate the performance of different classifiers various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, Matthews's Correlation Coefficient, and ROC Curve (AUC) are used. The architecture of the experimental framework is placed below in Figure-4.5.

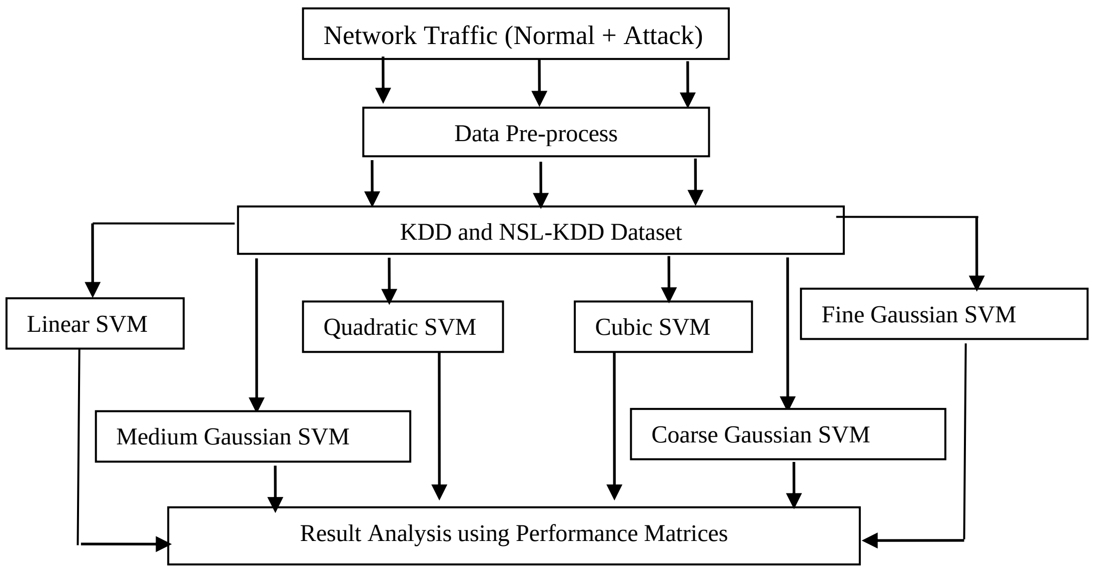


Fig. 4.5: Experimental Framework for Support Vector Machine

4.7 KDD Dataset for Support Vector Machine Experiments (training and test)

The KDD (training and test) dataset are analyzed after pre-processing and reducing redundant data, 25190 input instances are selected for experimentation purposes. Then trainset was divided into ten sets randomly, containing both normal and attack data that appears in the KDD dataset.

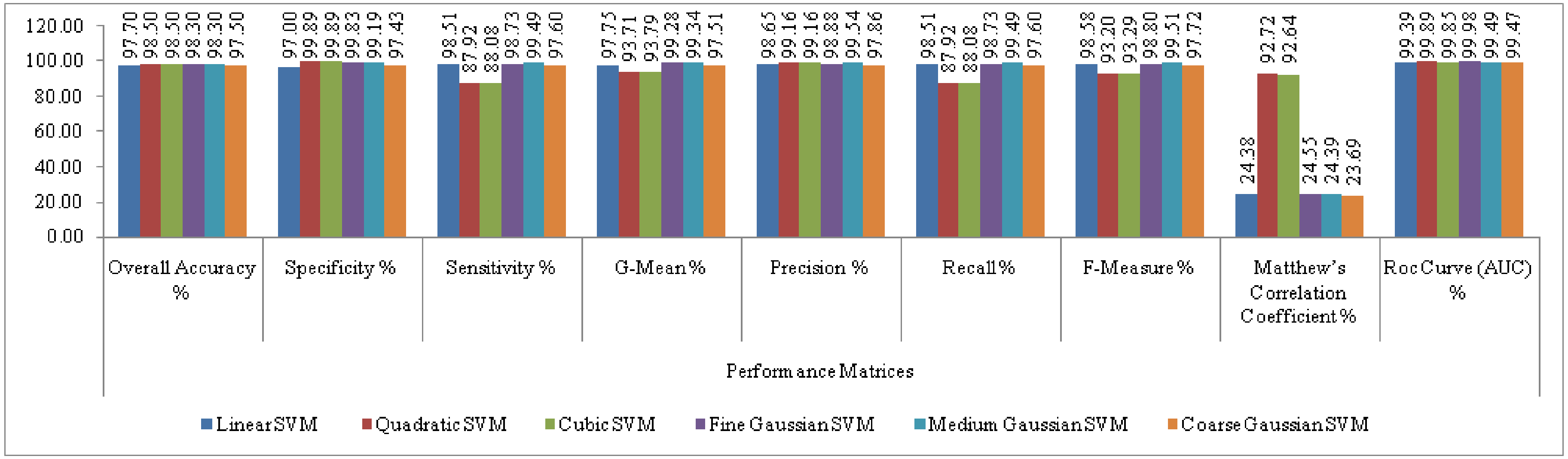
The different attacks contained in the KDD dataset are back, land, Neptune, pod, smurf, teardrop, satan, ipsweep, nmap, portsweep, guess_password, ftp_write, imap, phf, multihop, warezmaster, warezclient, spy, buffer_overflow, load-module, perl, and rootkit, etc. These all-attack types fall under four attack categories i.e., Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probe. Out of ten dataset, nine datasets were selected for training and one dataset was selected randomly for the test. The training dataset comprise total instances, 929511 (2519*41* 9=929511) while, the test dataset constitutes the total instances, 103279 (2519*41*1= 103279). The program itself randomly selects the data for training and testing purposes. The distribution of data for the KDD dataset is shown in Table-4.3.

Table 4.3- Distribution of data for KDD (training & test) experiments

Dataset Name	No. of Features	Input instances	Total
Trainset data 1	41	2519	2519* 41*1= 103279
Trainset data 2	41	2519	2519* 41*1= 103279
Trainset data 3	41	2519	2519* 41*1= 103279
Trainset data 4	41	2519	2519* 41*1= 103279
Trainset data 5	41	2519	2519* 41*1= 103279
Trainset data 6	41	2519	2519* 41*1= 103279
Trainset data 7	41	2519	2519* 41*1= 103279
Trainset data 8	41	2519	2519* 41*1= 103279
Trainset data 9	41	2519	2519* 41*1= 103279
	Total Training instances		**929511**
Testset data 1	41	2519	2519 *41*1= 103279
	Total Test instances		**103279**

Table- 4.4: Experimental Results of Support Vector Machine using KDD (training) dataset

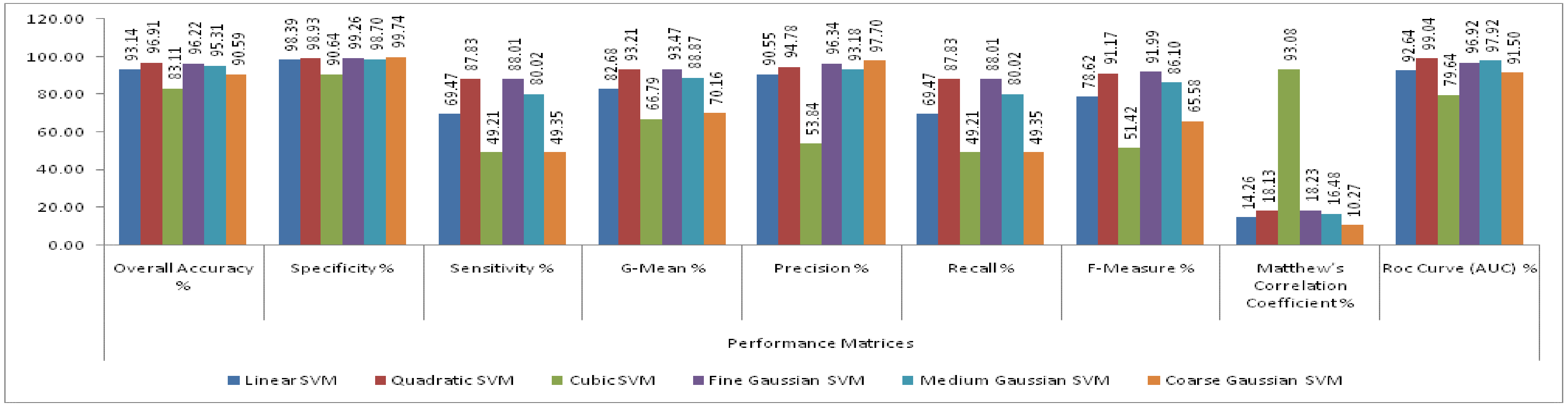
Classifiers	Performance Matrices								
	Overall Accuracy%	Specificity%	Sensitivity%	G-Mean%	Precision%	Recall%	F-Measure%	Matthew's Correlation Coefficient%	ROC Curve (AUC) %
Linear SVM	97.70	97.00	98.51	97.75	98.65	98.51	98.58	24.38	99.39
Quadratic SVM	98.50	99.89	87.92	93.71	99.16	87.92	93.20	92.72	99.89
Cubic SVM	98.50	99.89	88.08	93.79	99.16	88.08	93.29	92.64	99.85
Fine Gaussian SVM	98.30	99.83	98.73	99.28	98.88	98.73	98.80	24.55	99.98
Medium Gaussian SVM	98.30	99.19	99.49	99.34	99.54	99.49	99.51	24.39	99.49
Coarse Gaussian SVM	97.50	97.43	97.60	97.51	97.86	97.60	97.72	23.69	99.47



Graph-4.1: Experimental Results of Support Vector Machine using KDD (training) dataset

Table- 4.5: Experimental Results of Support Vector Machine using KDD (test) dataset

Classifiers	Performance Matrices								
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-Measure %	Matthew's Correlation Coefficient %	ROC Curve (AUC) %
Linear SVM	93.14	98.39	69.47	82.68	90.55	69.47	78.62	14.26	92.64
Quadratic SVM	96.91	98.93	87.83	93.21	94.78	87.83	91.17	18.13	99.04
Cubic SVM	83.11	90.64	49.21	66.79	53.84	49.21	51.42	93.08	79.64
Fine Gaussian SVM	96.22	99.26	88.01	93.47	96.34	88.01	91.99	18.23	96.92
Medium Gaussian SVM	95.31	98.70	80.02	88.87	93.18	80.02	86.10	16.48	97.92
Coarse Gaussian SVM	90.59	99.74	49.35	70.16	97.70	49.35	65.58	10.27	91.50



Graph-4.2: Experimental Results of Support Vector Machine using KDD (test) dataset

4.8 Experimental Results

The experimental results of six different linear and non-linear Support Vector Machine classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM are discussed. Table-4.4 shows the results of these classifiers for different performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, Matthews's Correlation Coefficient, and ROC Curve (AUC). The evaluated results for both training and test were obtained after proper validation of the experiments.

4.8.1 Experimental Results of KDD (training) and Analysis

The observational results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, Matthews's Correlation Coefficient and ROC Curve (AUC) for six different linear and non-linear Support Vector Machine classifiers i.e. Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian Support Vector Machine using KDD (Training) dataset as stated in section 4.7 are placed in Table-4.4 corresponding with Graph- 4.1.

The table on the analysis found that Quadratic SVM and Cubic SVM have performed the best result for Overall Accuracy i.e. 98.50% each as compared to other classifiers. It means that these classifiers accurately classify the training dataset as compared to another classifier. But the classifiers like Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 98.30% each followed by both Quadratic SVM and Cubic SVM, which shows very good performance. But the other classifiers like Linear SVM and Coarse Gaussian SVM also show good performance for the result of Overall Accuracy i.e. 97.70% and 97.50% respectively, while classifying the training dataset.

The result for Specificity or True Negative Rate (TNR), both Quadratic SVM and Cubic SVM performed the best result i.e. 99.89% each, which otherwise means that both Quadratic SVM and Cubic SVM correctly classify negative classes. But other classifiers like Fine Gaussian SVM and Medium Gaussian Support Vector

Machine classify the dataset with 99.83% and 99.19% correspondingly. But Coarse Gaussian SVM classifies the training instance with 97.43% and Linear Support Vector Machine classifies the training instance with 97.00%, which also shows very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate (TPR), Medium Gaussian Support Vector Machine performed the best result for Sensitivity (99.49%). It means that Medium Gaussian Support Vector Machine accurately classifies the true positive instances or positive classes of the present dataset whereas, Fine Gaussian SVM and Linear Support Vector Machine shows 98.73% and 98.51% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian Support Vector Machine performed 97.60%. It also shows a very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e. 88.08% and 87.92% respectively.

For the result of G-mean or G-measure, the Medium Gaussian SVM performed the best result i.e. 99.34%, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (99.19%) and Sensitivity (99.49%) of Medium Gaussian SVM are high then the G-mean of Medium Gaussian SVM is also relatively high. Further, the G-mean value for Fine Gaussian Support Vector Machine constitutes 99.28%, because the value of both Specificity (99.83%) and Sensitivity (98.73%) are also superior, which is slightly fewer performance than Medium Gaussian SVM. But the performances of the other classifiers like Linear SVM (97.75%) and Coarse Gaussian SVM (97.51%) also show good performance. Then finally Cubic SVM and Quadratic SVM also illustrate good performance as compared to others. i.e. 93.79% and 93.71% correspondingly.

The result of Precision or Positive Predictive Value, Medium Gaussian SVM performed the best result i.e. 99.54% of the exactness or quality of the classifier. But the other classifiers like Cubic SVM and Quadratic SVM, both show 99.16% of exactness. Then Fine Gaussian SVM and Linear SVM show 98.88% and 98.65%of exactness. Finally, Coarse Gaussian SVM (97.86%), which shows slightly fewer performance than other classifiers.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate (TPR), again Medium Gaussian SVM performed the best result i.e. 99.49%. It means that Medium Gaussian SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, the other classifiers like Fine Gaussian SVM (98.53%) and Linear SVM (98.51%), which is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. But the other classifiers like Coarse Gaussian SVM (97.60%) also show very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e. 88.08% and 87.92% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here again, Medium Gaussian SVM performed the best result i.e. 99.51% because the result for both Precision and Recall is relatively very high. i.e. 99.54% and 99.49% respectively. Then the result for Fine Gaussian SVM (98.80%) shows a very good result because the result for both Precision and Recall also shows very good i.e. 98.88% and 98.73%. Then the result of Linear SVM comes with 98.58% which also shows very good performance, because of the result of Precision i.e. 98.65%, and Recall i.e. 98.51% which is a comparatively very good performance. But the classifier like Coarse Gaussian SVM shows 97.72%. It also shows a very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e.93.29% and 93.20% respectively because the Recall value of both the classifiers is 88.08% and 87.92% respectively.

The result of MCC for both Quadratic SVM and Cubic SVM shows the best result among all other algorithms i.e. 92.72%and 92.64% respectively, whereas, the other classifiers shows very low performance i.e. Fine Gaussian SVM (24.55%), Medium Gaussian SVM (24.39%), Linear SVM (24.38%) and finally Coarse Gaussian SVM shows 23.69% for MCC, which is considered as very low performance among other classifiers.

The Receiver Operating Characteristic curve (ROC) and its Area under the Curve (AUC) are used for the performance analysis of the classifier. The ROC graph is the trade-off between benefits and costs. Here, ‘1’ is the best possible value. The Fine Gaussian SVM is shown preeminent results among all the classifiers i.e. 99.98%. The ROC Curve (AUC) for different linear and non-linear Support Vector Machine classifiers is given below in Figure-4.6 to Figure-4.11.

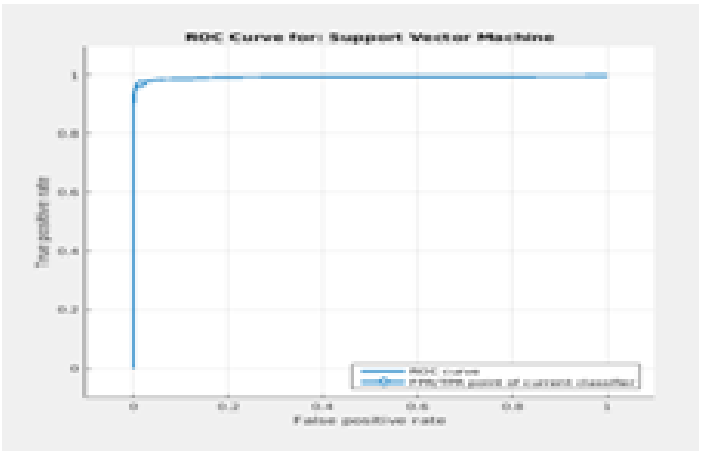


Fig. 4.6: ROC Curve for Linear SVM

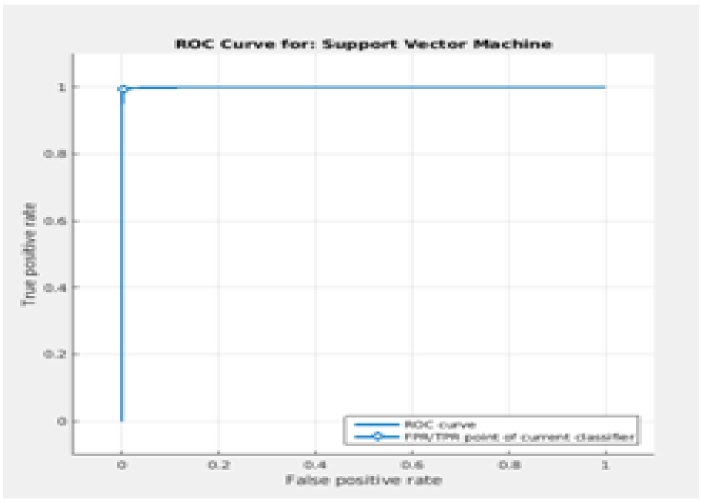


Fig. 4.7: ROC Curve for Quadratic SVM

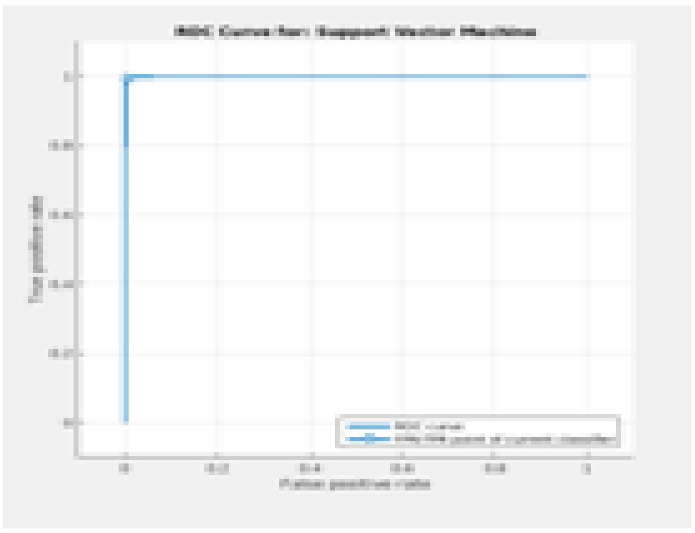


Fig. 4.8: ROC Curve for Cubic SVM

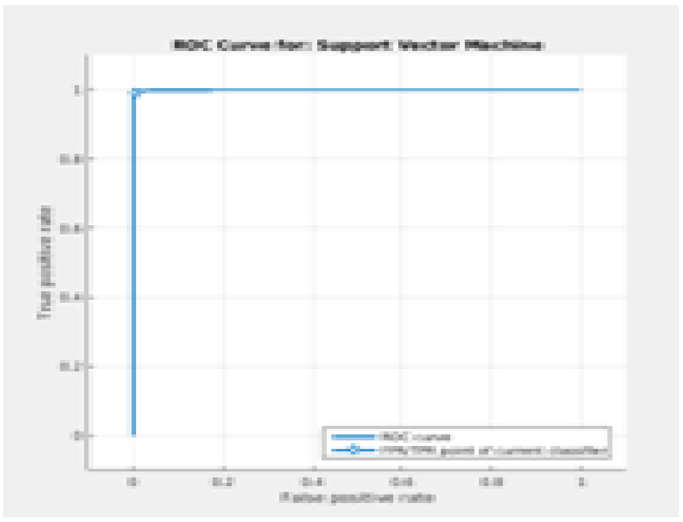


Fig.4.9: ROC Curve for Medium Gaussian SVM

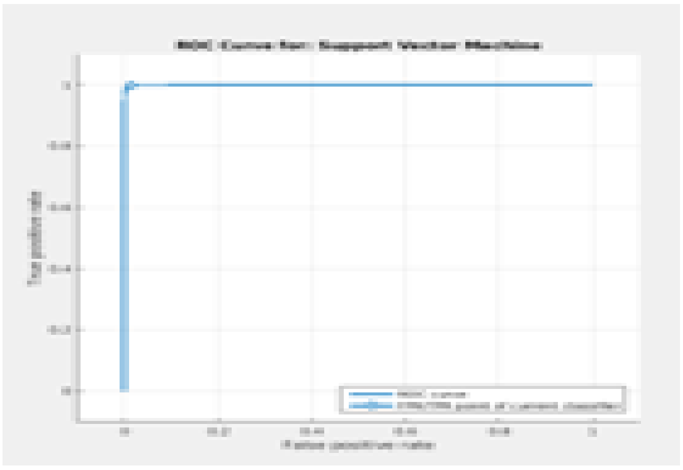


Fig. 4.10: ROC Curve for Fine Gaussian SVM

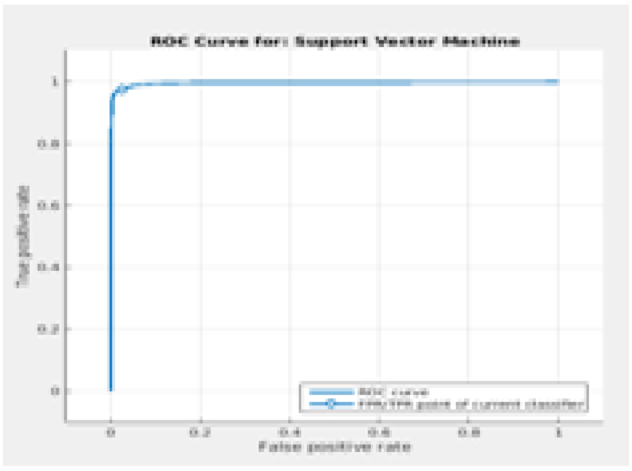


Fig. 4.11: ROC Curve for Coarse Gaussian SVM

4.8.2 Experimental Results of KDD (test) and Analysis

The observational results of six different linear and non-linear Support Vector Machine classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM using KDD (Test) dataset as stated in section 4.7 are placed in Table-4.5 corresponding with Graph- 4.2. The table shows the results of these classifiers for different performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, Matthews's Correlation Coefficient, and ROC Curve (AUC).

Table- 4.5 shows that Quadratic SVM has performed the best result for Overall Accuracy i.e. 96.91%as compared to other classifiers. It means that this classifier accurately classifies the test dataset as compared to other classifiers. But the Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 96.22% and 95.31% respectively followed by Quadratic SVM, which shows very good performance. But the other classifiers like Linear SVM, Coarse Gaussian SVM, and Cubic SVM also show good performance for the result of Overall Accuracy i.e. 93.14%, 90.59%, and 83.11% respectively, while classifying the test dataset.

The result for Specificity or True Negative Rate (TNR), Coarse Gaussian SVM has performed the best result i.e. 99.74%, which otherwise means that Coarse Gaussian SVM correctly classifies negative classes. But other classifiers like Fine

Gaussian SVM classify the dataset with 99.26%, which also shows very good performance. But Quadratic SVM classifies the instance with 98.93% followed by Medium Gaussian SVM, which shows the result with 98.70%. Then other classifiers like Linear SVM and Cubic SVM reveal the result with 98.39% and 90.64% respectively, which is also considered as very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate (TPR), Fine Gaussian SVM performed the best result for Sensitivity is 88.01%. It means that Fine Gaussian SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Quadratic SVM and Medium Gaussian SVM shows 87.83% and 80.02% respectively and it is also considered as better performance for Sensitivity as compared with Fine Gaussian SVM. Then Coarse Gaussian SVM performed 97.60%. It also shows a very good performance. Finally, Linear SVM, Coarse Gaussian SVM, and Cubic SVM show very low performance as compared to others. i.e. 69.47%, 49.35% and 49.21% respectively.

For the result of G-mean or G-measure, the Fine Gaussian SVM performed the best result i.e. 93.47%, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (99.26%) and Sensitivity (88.01%) are high. So, that the G-mean of Fine Gaussian SVM is also relatively high. Further, the G-mean value for Quadratic SVM constitutes 93.21%, because the value of both Specificity (98.93%) and Sensitivity (87.83%) are also superior, which is slightly less performance than Fine Gaussian SVM. But the performances of the other classifiers like Medium Gaussian SVM (88.87%) and Linear SVM (82.68%) also show good performance. Then finally Coarse Gaussian SVM and Cubic SVM reveal not very good performance as compared to others. i.e. 70.16% and 66.79% correspondingly.

The result of Precision or Positive Predictive Value, Coarse Gaussian SVM performed the best result i.e. 97.70%of exactness or quality of the classifier followed by Fine Gaussian Support Vector Machine which is constituted with 96.34% of exactness. But the other classifiers like Quadratic SVM and Medium Gaussian SVM both show 94.78% and 93.18% of exactness respectively. Then Linear SVM and

Cubic SVM show 90.55% and 53.84%of exactness. So, finally, Cubic SVM shows very low performance for Precision than other classifiers.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate (TPR), Fine Gaussian SVM performed the best result for Sensitivity is 88.01%. It means that Fine Gaussian SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Quadratic SVM and Medium Gaussian SVM shows 87.83% and 80.02% respectively and it is also considered as better performance for Sensitivity as compared with Fine Gaussian SVM. Then Coarse Gaussian SVM performed 97.60%. It also shows a very good performance. Finally, Linear SVM, Coarse Gaussian SVM, and Cubic SVM show very low performance as compared to others. i.e. 69.47%, 49.35% and 49.21% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Fine Gaussian SVM performed the best result i.e. 91.99% because the result for both Precision and Recall is relatively very high. i.e. 96.34% and 88.01% respectively. Then the result for Quadratic SVM shows very good performance followed by Fine Gaussian SVM i.e.91.17% because the result for both Precision and Recall also shows very good i.e. 94.78% and 87.83%. The result of Medium Gaussian SVM comes with 86.10%, which is also considered a very good performance because the result of Precision is 93.18% and Recall is 80.02%, which is a comparatively very good performance. But the classifier like Linear SVM shows 78.62% due to its corresponding Recall value i.e. 69.47%, which shows low performance. Finally, Coarse Gaussian SVM and Cubic SVM both are show slightly low performance as compared to others. i.e. 65.58% and 51.42% respectively, because the Recall value of both the classifiers is 49.35% and 49.21% respectively, which are considered as very low performance.

The result of MCC for Cubic SVM shows the best result among all other algorithms i.e. 93.08%, whereas, the other classifiers show very low performance i.e.

Fine Gaussian SVM (18.23%), Quadratic SVM (18.13%), Medium Gaussian SVM (16.48%), Linear SVM (14.26%) and finally Coarse Gaussian SVM shows 10.27% for MCC, which is considered as very low performance among other classifiers.

The Receiver Operating Characteristic curve (ROC) and its Area under the Curve (AUC) are used for the performance analysis of the classifier. The ROC graph is the trade-off between benefits and costs. Here, '1' is the best possible value. The Quadratic SVM is shown preeminent results among all the classifiers i.e. 99.04%. The ROC Curve (AUC) for different linear and non-linear Support Vector Machine classifiers is given below in Figure-4.12 to Figure-4.17.

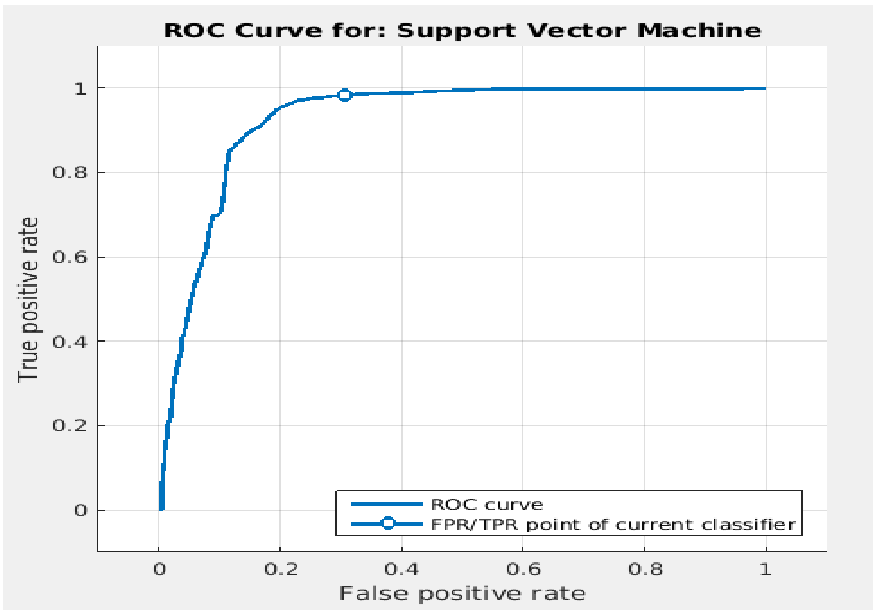


Fig. 4.12: ROC Curve for Linear SVM

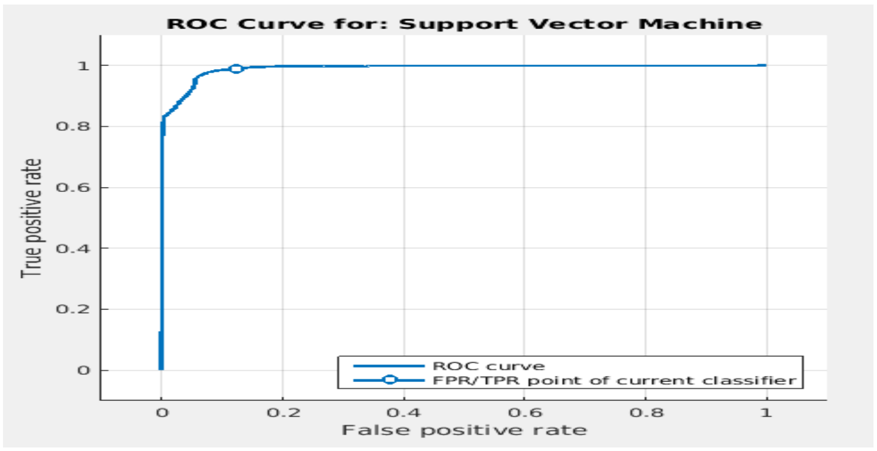


Fig.4.13: ROC Curve for Quadratic SVM

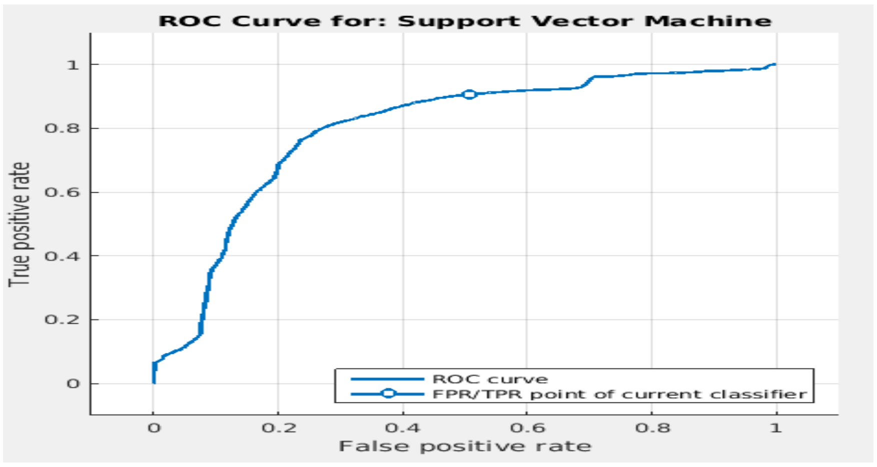


Fig. 4.14: ROC Curve for Cubic SVM

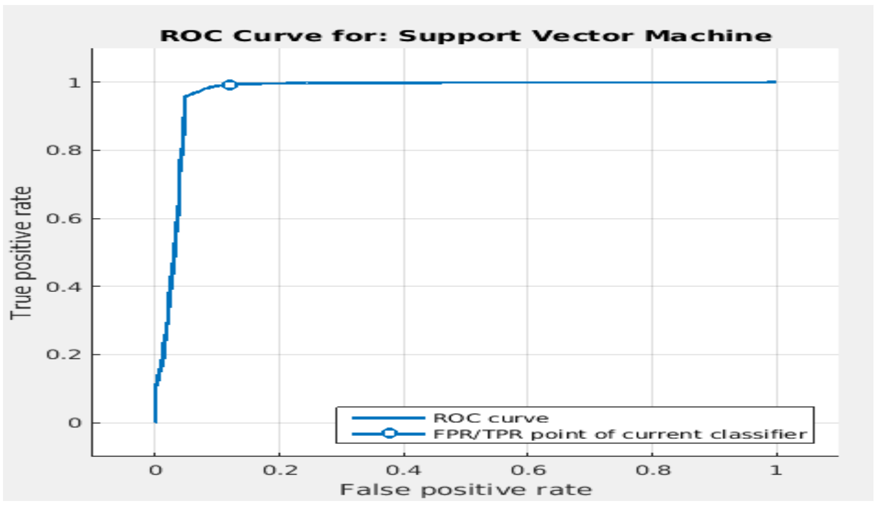


Fig. 4.15: ROC Curve for Fine Gaussian SVM

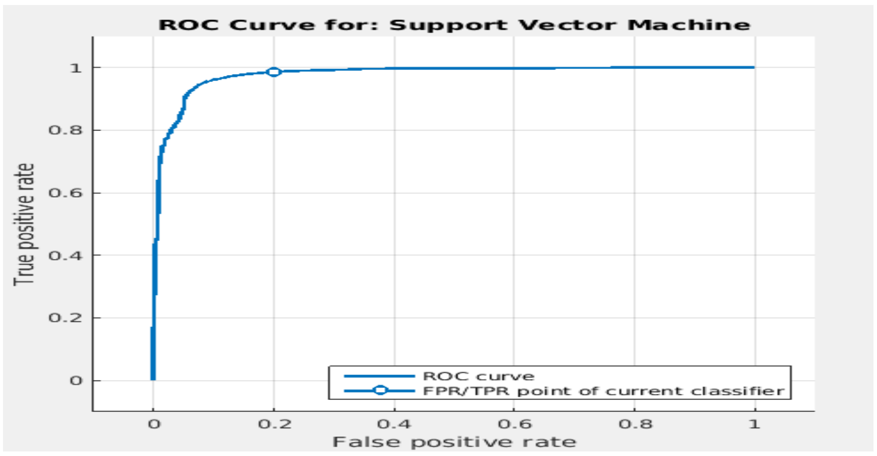


Fig. 4.16: ROC Curve for Medium Gaussian SVM

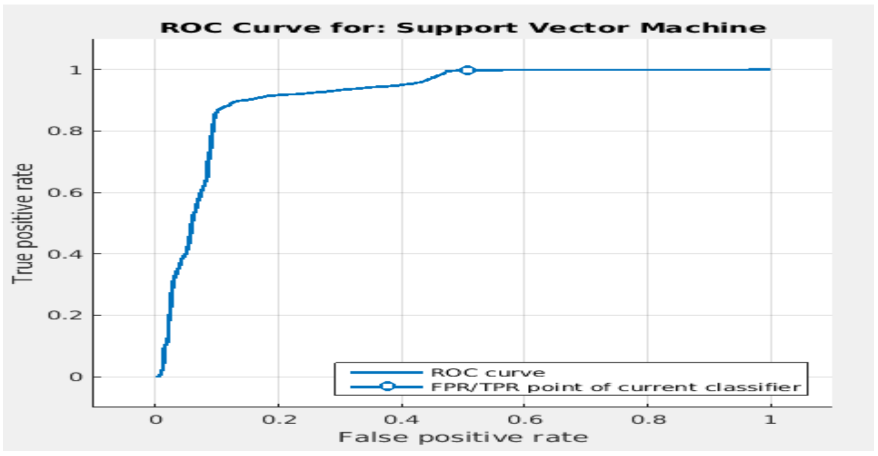


Fig.4.17: ROC Curve for Coarse Gaussian SVM

4.8.3 Description of various Parameters after Training

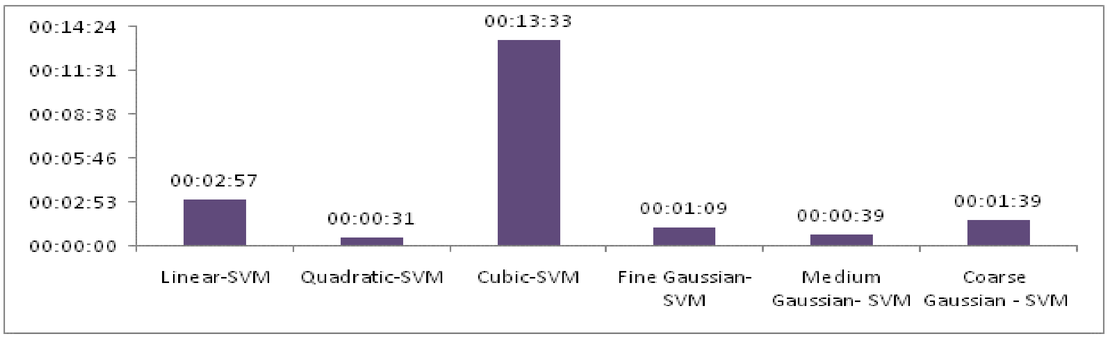
Different classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM have been used for the classification of the KDD dataset. Table- 4.6 shows the parameters of different classifiers after train the dataset.

Table- 4.6: Description of Parameters after Training

A	Linear SVM:	Training time: 00:02:57 Classifier options: type = SVM, kernel function: linear, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
B	Quadratic SVM:	Training time: 00:00:31 Classifier options: type = SVM, kernel function: Quadratic, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
C	Cubic SVM:	Training time: 00:13:33 Classifier options: type = SVM, kernel function: cubic, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
D	Fine Gaussian	Training time: 00:01:09 Classifier options: type = SVM, kernel function: Gaussian, manual

	SVM:	kernel scale = 1.6, kernel scale mode = manual, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
E	Medium Gaussian SVM:	Training time: 00:00:39 Classifier options: type = SVM, kernel function: Gaussian, manual kernel scale = 6.4, kernel scale mode = manual, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
F	Coarse Gaussian SVM:	Training time: 00:01:39 Classifier options: type = SVM, kernel function: Gaussian, manual kernel scale = 26.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41

Further, the training time of the different linear and nonlinear kernel SVMs is depicted in Graph-4.3. Table-4.6 expresses the parameters along with the corresponding training time of different linear and nonlinear kernel SVMs after train the dataset. From this table, we deduced that Quadratic SVM is taking the optimal time i.e., 00:00:31 to train the dataset.



Graph-4.3: Training time of the linear and nonlinear kernel SVMs

4.9 NSL-KDD Dataset for Support Vector Machine Experiments (training and test)

The NSL-KDD (training and test) dataset are analyzed after pre-processing and reducing redundant data, 25190 input instances are selected for experimentation purposes. Then trainset was divided into ten sets randomly, containing both normal and attack data that appears in the NSL-KDD dataset.

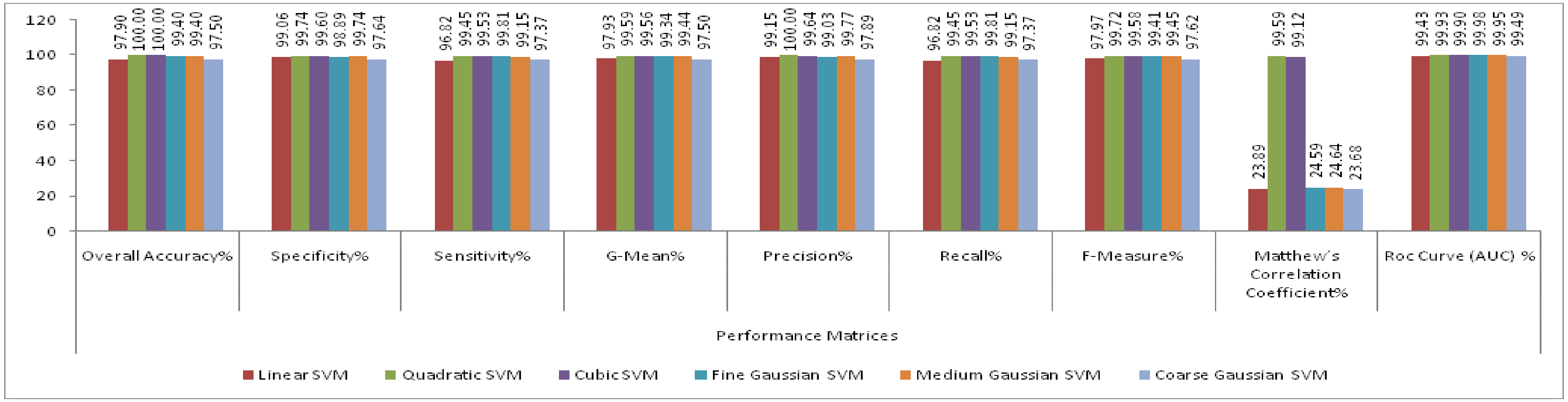
The different attacks contained in the NSL-KDD dataset are back, land, Neptune, pod, smurf, teardrop, satan, ipsweep, nmap, portsweep, guess_password, ftp_write, imap, phf, multihop, warezmaster, warezclient, spy, buffer_overflow, load-module, snmpgetattack, xlock, sendmail, apache2, udpstorm, xsnoop , xterm, mscan , processtable, httptunnel, mailbomb , ps, snapgueuss, named, saint, perl, and rootkit, etc. These all attack types fall under four attack categories i.e., Denial of Service (Dos), User to Root (U2R), Remote to Local (R2L), and Probe. Out of ten dataset, nine dataset were selected for training and one dataset was selected randomly for the test. The training dataset comprise total instances, 929511 (2519*41* 9=929511) while, the test dataset constitutes the total instances, 103279 (2519*41= 103279). The program itself randomly selects the data for training and testing purposes. The distribution of data for the NSL-KDD dataset is shown in Table-4.7.

Table-4.7: Distribution of data for NSL-KDD (training & test) experiments

Dataset Name	No. of Features	Input instances	Total
Trainset data 1	41	2519	2519* 41*1= 103279
Trainset data 2	41	2519	2519* 41*1= 103279
Trainset data 3	41	2519	2519* 41*1= 103279
Trainset data 4	41	2519	2519* 41*1= 103279
Trainset data 5	41	2519	2519* 41*1= 103279
Trainset data 6	41	2519	2519* 41*1= 103279
Trainset data 7	41	2519	2519* 41*1= 103279
Trainset data 8	41	2519	2519* 41*1= 103279
Trainset data 9	41	2519	2519* 41*1= 103279
	Total Training instances		**929511**
Testset data 1	41	2519	2519 *41*1= 103279
	Total Test instances		**103279**

Table-4.8: Experimental Results of Support Vector Machine using NSL-KDD (training) dataset

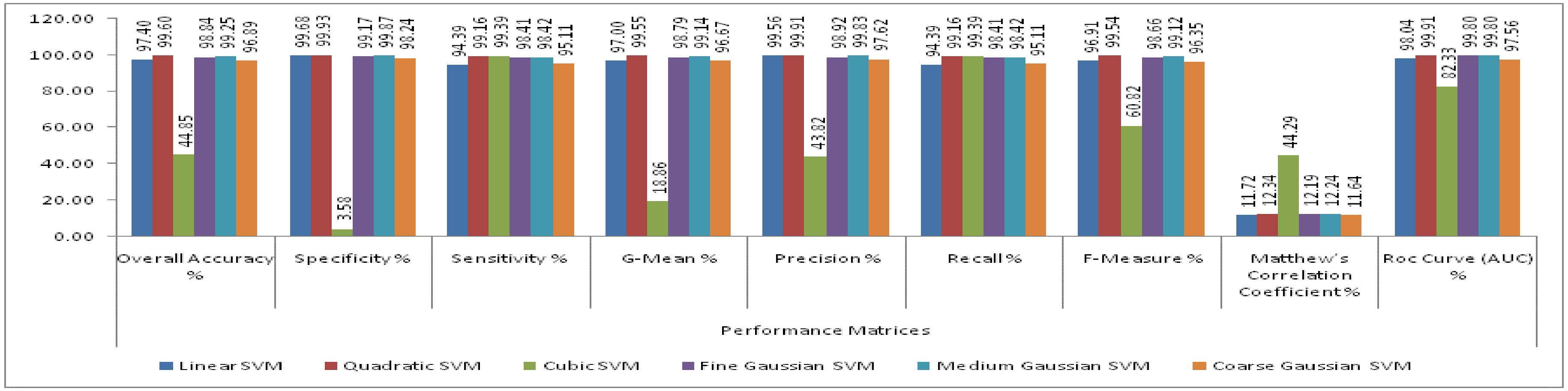
Classifiers	Performance Matrices								
	Overall Accuracy%	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-Measure %	Matthew's Correlation Coefficient%	ROC Curve (AUC) %
Linear SVM	97.90	99.06	96.82	97.93	99.15	96.82	97.97	23.89	99.43
Quadratic SVM	100.00	99.74	99.45	99.59	100.00	99.45	99.72	99.59	99.93
Cubic SVM	100.00	99.60	99.53	99.56	99.64	99.53	99.58	99.12	99.90
Fine Gaussian SVM	99.40	98.89	99.81	99.34	99.03	99.81	99.41	24.59	99.98
Medium Gaussian SVM	99.40	99.74	99.15	99.44	99.77	99.15	99.45	24.64	99.95
Coarse Gaussian SVM	97.50	97.64	97.37	97.50	97.89	97.37	97.62	23.68	99.49



Graph 4.4: Experimental Results of Support Vector Machine using NSL-KDD (training) dataset

Table-4.9: Experimental Results of Support Vector Machine using NSL-KDD (test) dataset

Classifiers	Performance Matrices								
	Overall Accuracy %	Specificity %	Sensitivity %	G-Mean %	Precision %	Recall %	F-Measure %	Matthews's Correlation Coefficient %	ROC Curve (AUC) %
Linear SVM	97.40	99.68	94.39	97.00	99.56	94.39	96.91	11.72	98.04
Quadratic SVM	99.60	99.93	99.16	99.55	99.91	99.16	99.54	12.34	99.91
Cubic SVM	44.85	03.58	99.39	18.86	43.82	99.39	60.82	44.29	82.33
Fine Gaussian SVM	98.84	99.17	98.41	98.79	98.92	98.41	98.66	12.19	99.80
Medium Gaussian SVM	99.25	99.87	98.42	99.14	99.83	98.42	99.12	12.24	99.80
Coarse Gaussian SVM	96.89	98.24	95.11	96.67	97.62	95.11	96.35	11.64	97.56



Graph-4.5- Experimental Results of Support Vector Machine using NSL-KDD (test) dataset

4.10 Experimental Results

In this section, the experimental results of linear and non-linear Support Vector Machine classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM are discussed. To evaluate the performance of these algorithms different performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, Matthews's Correlation Coefficient, and ROC Curve are used. The evaluated results for both training and test are obtained after proper validation of the experiments.

4.10.1 Experimental Results of NSL-KDD (training) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, Matthews's Correlation Coefficient and ROC Curve for six different linear and non-linear Support Vector Machine classifiers using NSL-KDD (training) dataset as stated in section 4.9 is placed in Table- 4.8 with the corresponding Graph- 4.4.

The table shows that Quadratic SVM and Cubic SVM have performed the best result for Overall Accuracy i.e. 100% each as compared to other classifiers. It means that these classifiers accurately classify the training dataset as compared to another classifier. But the Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 99.40% each, which shows very good performance. But the other classifiers like Linear SVM and Coarse Gaussian SVM also show good performance for the result of Overall Accuracy i.e. 97.90% and 97.50% respectively, while classifying the training dataset.

The result for Specificity or True Negative Rate, both Quadratic SVM and Medium Gaussian SVM performed the best result i.e. 99.74% each, which otherwise means that both Quadratic SVM and Medium Gaussian SVM correctly classify the negative classes. But other classifiers like Cubic SVM and Linear SVM classify the dataset with 99.60% and 99.06% correspondingly, which is also considered as very good performance. But, Fine Gaussian SVM and Coarse Gaussian SVM classify the training instance with 98.89% and 97.64% respectively, which also shows very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Fine Gaussian SVM performed the best result for Sensitivity (99.81%). It means that Fine Gaussian SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Cubic SVM, Quadratic SVM, and Medium Gaussian SVM show 99.53%, 99.45%, and 99.15% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian SVM performed 97.37%. It also shows a very good performance. Finally, Linear SVM also shows good performance as compared to others. i.e.96.82%.

For the result of G-mean or G-measure, the Quadratic SVM performed the best result i.e. 99.59%, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (99.74%) and Sensitivity (99.45%) of Quadratic SVM are high then the G-mean of Quadratic SVM (99.59%) is also relatively high. Further, the G-mean value for Cubic SVM constitutes 99.56%, because the value of both Specificity (99.60%) and Sensitivity (99.53%) are also superior, which is slightly less performance than Quadratic SVM. But the performances of the other classifiers like Medium Gaussian SVM (99.44%) and Fine Gaussian SVM (99.34%) also show very good performance. Then finally Linear SVM and Coarse Gaussian SVM also illustrate good performance as compared to others. i.e. 97.93% and 97.50% correspondingly.

The result of Precision or Positive Predictive Value, Quadratic SVM performed the best result i.e. 100%of exactness or quality of the classifier. But the classifiers like Medium Gaussian SVM and Cubic SVM also show the best result i.e. 99.77% and 99.64% of exactness or quality respectively as compared to Quadratic SVM. Then Linear SVM and Fine Gaussian SVM show 99.15% and 99.03%of exactness, which is considered as the better result for Precision. Finally, Coarse Gaussian SVM (97.89%), which shows slightly fewer performance than other classifiers.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or

Sensitivity or True Positive Rate (TPR), Fine Gaussian SVM performed the best result for Sensitivity (99.81%). It means that Fine Gaussian SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Cubic SVM, Quadratic SVM, and Medium Gaussian SVM show 99.53%, 99.45%, and 99.15% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian SVM performed 97.37%. It also shows a very good performance. Finally, Linear SVM also shows good performance as compared to others. i.e. 96.82%.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Quadratic SVM performed the best result i.e. 99.72% because the result for both Precision and Recall is relatively very high. i.e. 100% and 99.45% respectively. The result for Cubic SVM (99.58%) shows very good performance because the result for both Precision and Recall also shows very good i.e. 99.64% and 99.53% respectively. Then the result of Medium Gaussian SVM comes with 99.45%, which is also considered as very good performance, because of the result of Precision (99.77%) and Recall (99.15%), which is a comparatively very good performance. But the classifier like Fine Gaussian SVM shows 99.41% followed by Medium Gaussian SVM. It also reveals very good results for F-measure. Finally, Linear SVM and Coarse Gaussian SVM also show good performance as compared to others. i.e.97.97% and 97.62% respectively because the Precision value of both the classifier is very good i.e. 99.15% and 97.89% respectively.

The result of MCC for Quadratic SVM shows the best result i.e. 99.59% followed by Cubic SVM, which is also showing very good result i.e. 99.12% whereas, the other classifiers show very low performance i.e. Medium Gaussian SVM (24.64%), Fine Gaussian SVM (24.59%) followed by both Linear SVM and Coarse Gaussian SVM i.e. 23.89% and 23.68% respectively, which is considered as very low performance as compared to both Quadratic SVM and Cubic Support Vector Machine.

The Receiver Operating Characteristic curve (ROC) and its Area under the Curve (AUC) are used for the performance analysis of the classifier. The ROC graph is the trade-off between benefits and costs. Here, '1' is the best possible value. The Fine Gaussian SVM is shown preeminent results among all the classifiers i.e. 99.98%. The ROC Curve (AUC) for different linear and non-linear Support Vector Machine classifiers is shown below in Figure-4.18 to 4.23.

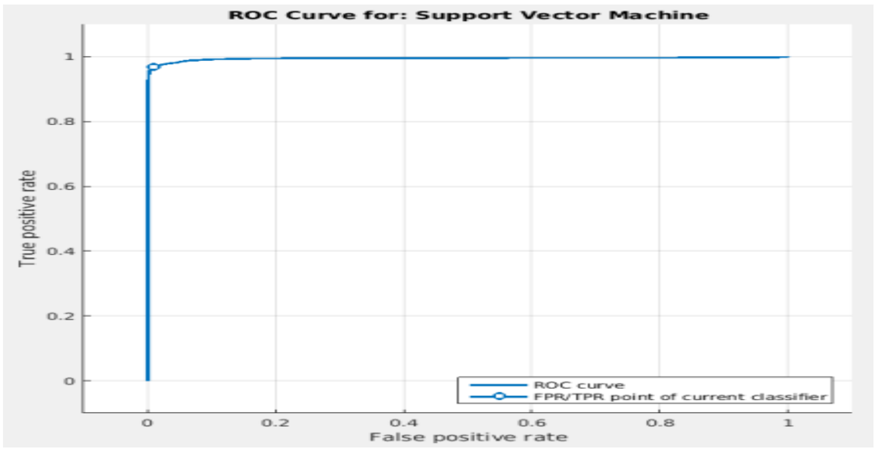


Fig. 4.18: ROC Curve for Linear SVM

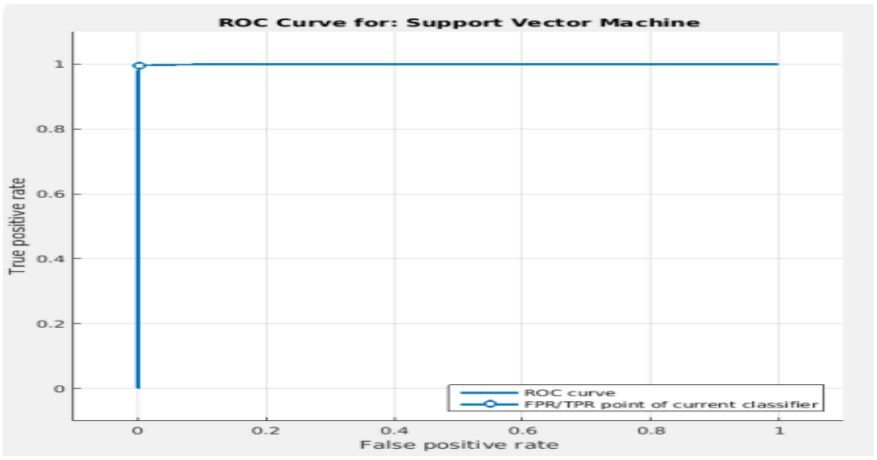


Fig. 4.19: ROC Curve for Quadratic SVM

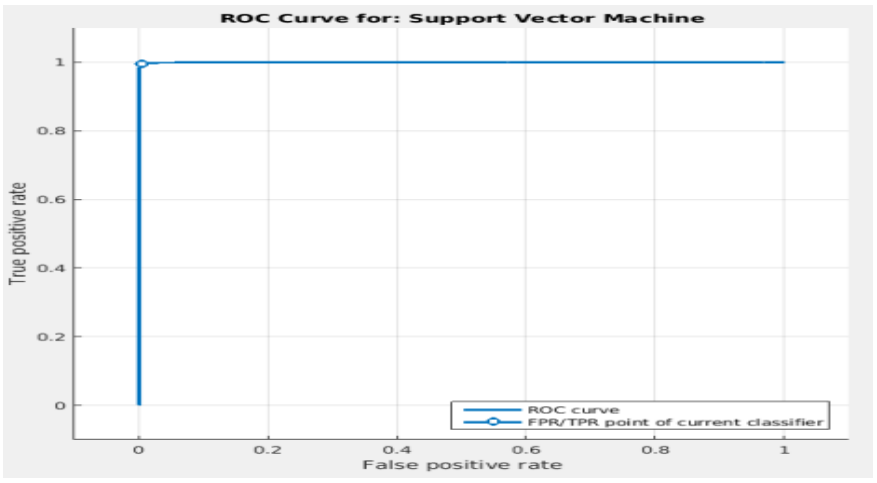


Fig. 4.20: ROC Curve for Cubic SVM

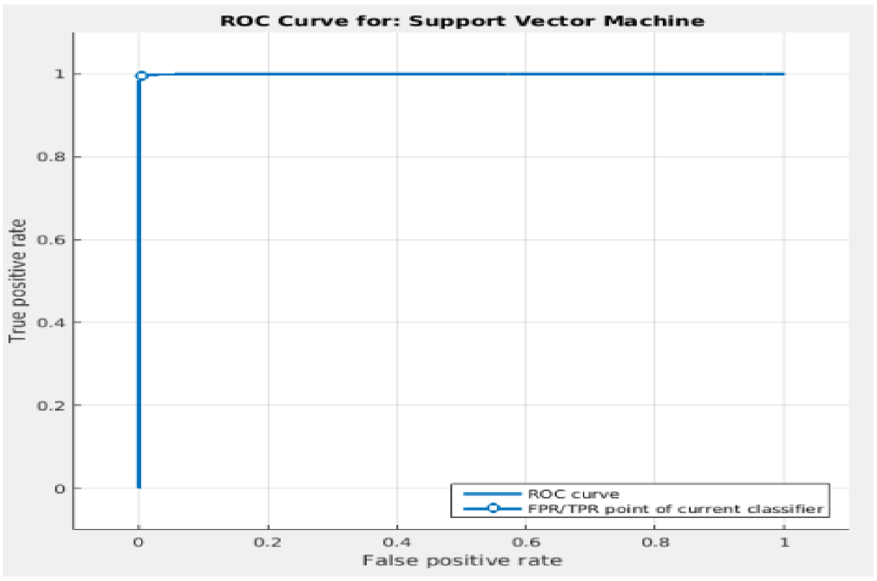


Fig. 4.21: ROC Curve for Fine Gaussian SVM

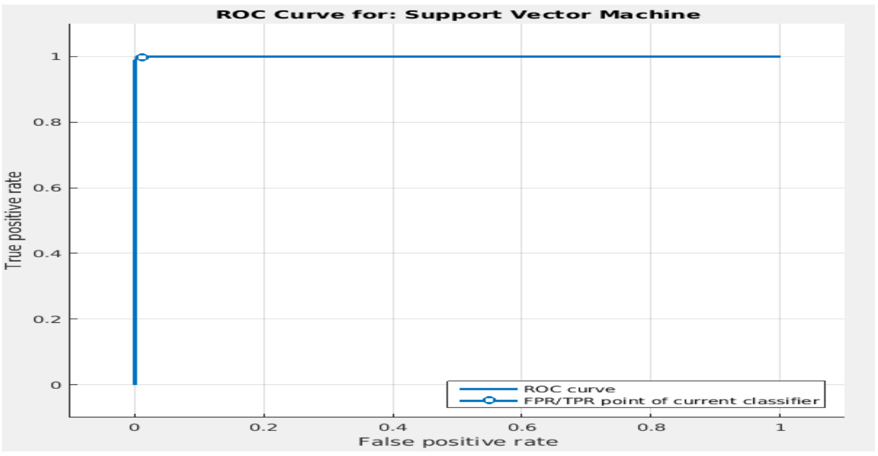


Fig. 4.22: ROC Curve for medium Gaussian SVM

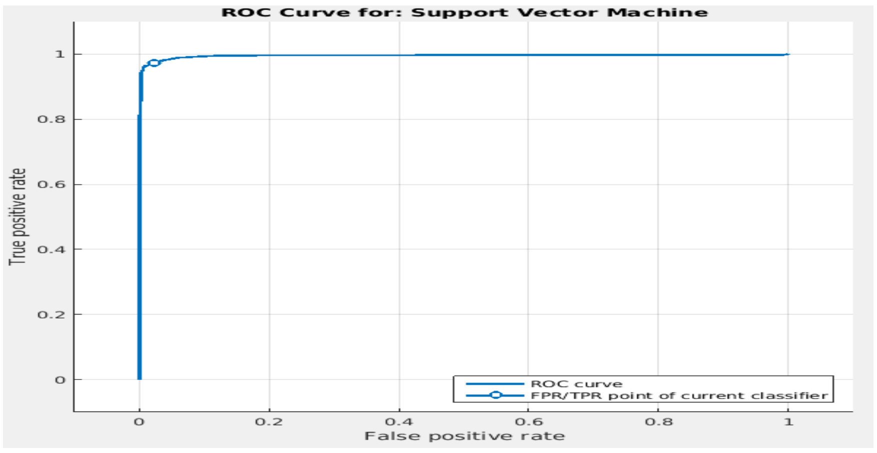


Fig. 4.23: ROC Curve for Coarse Gaussian SVM

4.10.2 Experimental Results of NSL-KDD (test) and Analysis

The experimental results for different performance matrices i.e, Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, Matthews's Correlation Coefficient and ROC Curve for six different linear and non-linear Support Vector Machine classifiers using NSL-KDD (test) dataset as stated in section 4.9 is placed in Table- 4.9 with the corresponding Graph- 4.5.

The table on the analysis found that Quadratic SVM has performed the best result for Overall Accuracy i.e. 99.60% as compared to other classifiers. It means that these classifiers accurately classify the test dataset as compared to other classifiers. But the Medium Gaussian SVM and Fine Gaussian SVM classify the instance with 99.25% and 98.84% respectively, which shows very good performance. But the other classifiers like Linear SVM, Coarse Gaussian SVM, and Cubic SVM also show good performance for the result of Overall Accuracy i.e. 97.40%, 96.89%, and 44.85% respectively, while classifying the test dataset. As per the result, Cubic SVM shows very low performance.

The result for Specificity or True Negative Rate, Quadratic SVM has performed the best result i.e. 99.74% each, which otherwise means that Quadratic SVM correctly classifies the negative classes of the present dataset. Then Medium Gaussian SVM classifies the negative classes with 99.87%, which is also considered as very good performance. Then the classifiers like Linear SVM and Fine Gaussian

SVM classify the dataset with 99.68% and 99.17% correspondingly, which is also considered as very good performance. But Coarse Gaussian SVM and Cubic SVM classify the test instance with 98.24% and 03.58% respectively. It means that Cubic SVM shows very low performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Cubic SVM performed the best result for Sensitivity (99.39%). It means that Cubic SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Quadratic SVM shows 99.16% result for Sensitivity, which is considered as very good performance as compared with Cubic SVM. Then Medium Gaussian SVM and Fine Gaussian SVM performed 98.42% and 98.41% respectively which also shows very good performance. Finally, Coarse Gaussian SVM and Linear SVM also show good performance as compared to others. i.e.95.11% and 94.39% respectively.

For the result of G-mean or G-measure, the Quadratic SVM performed the best result i.e. 99.55%, because it is a geometric mean of both Specificity and Sensitivity. So, here both Specificity (99.93%) and Sensitivity (99.16%) of Quadratic SVM are high then the G-mean of Quadratic SVM is also relatively high. Further, the G-mean value for Medium Gaussian SVM constitutes 99.14%, because the value of both Specificity (99.87%) and Sensitivity (98.42%) are also superior, which is slightly fewer performance than Quadratic SVM. But the performances of the other classifiers like Fine Gaussian SVM (98.79%) and Linear SVM (97.00%) also show very good performance. Then finally Coarse Gaussian SVM illustrates good performance as compared to others. i.e. 96.67%. But the Cubic SVM shows very low performance than others i.e. 18.86% because its Specificity value is very low i.e.03.58%.

The result of Precision or Positive Predictive Value, Quadratic SVM performed the best result i.e. 99.91%of the exactness or quality of the classifier. But the classifiers like Medium Gaussian SVM and Linear SVM also show a very good performance i.e. 99.83% and 99.56% of exactness or quality respectively as compared to Quadratic SVM. Then Fine Gaussian SVM and Coarse Gaussian SVM show 98.92% and 97.62%of exactness, which is also considered as the better result for Precision. Finally, Cubic SVM comes with 43.82%, which shows very low performance than other classifiers.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate, Cubic SVM performed the best result for Sensitivity (99.39%). It means that Cubic SVM accurately classifies the true positive instances or positive classes of the present dataset whereas, Quadratic SVM shows 99.16% result for Sensitivity, which is considered as very good performance as compared with Cubic SVM. Then Medium Gaussian SVM and Fine Gaussian SVM performed 98.42% and 98.41% respectively which also shows very good performance. Finally, Coarse Gaussian SVM and Linear SVM also show good performance as compared to others. i.e. 95.11% and 94.39% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Quadratic SVM performed the best result i.e. 99.54% because the result for both Precision and Recall is relatively very high. i.e. 99.91% and 99.16% respectively. Then the result for Medium Gaussian SVM is 99.12%, which shows very good performance because the result for both Precision and Recall also shows very good i.e. 99.83% and 98.42%. Then the result of Fine Gaussian SVM comes with 98.66%, which is also considered as very good performance, because of the result of Precision (98.92%) and Recall (98.41%), which is a comparatively very good performance. Then the classifier like Coarse Gaussian SVM and Linear SVM comes with 96.35% and 96.91% results respectively. It also reveals a very good result for F-measure. Finally, Cubic SVM shows low performance as compared to others. i.e. 60.82% because of its Precision value i.e. 43.82%.

The result of MCC for Cubic SVM shows the best result i.e.44.29%. But the other classifiers show very low performance as compared to Cubic SVM, i.e. Quadratic SVM (12.34%), Medium Gaussian SVM (12.24%), and Fine Gaussian SVM (12. 19%). Then finally both Linear SVM and Coarse Gaussian SVM show very low-performance i.e.11.72% and 11.64% respectively.

The Receiver Operating Characteristic curve (ROC) and its Area under the Curve (AUC) are used for the performance analysis of the classifier. The ROC graph is the trade-off between benefits and costs. Here, '1' is the best possible value. The Quadratic SVM is shown preeminent results among all the classifiers i.e. 99.91%. The ROC Curve (AUC) for different linear and non-linear Support Vector Machine classifiers is given below in Figure 4.24 to 4.29.

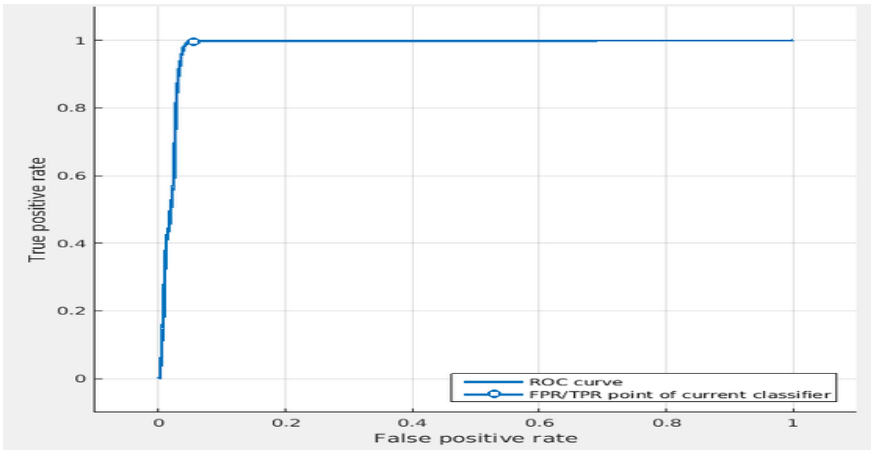


Fig. 4.24: ROC Curve for Linear SVM

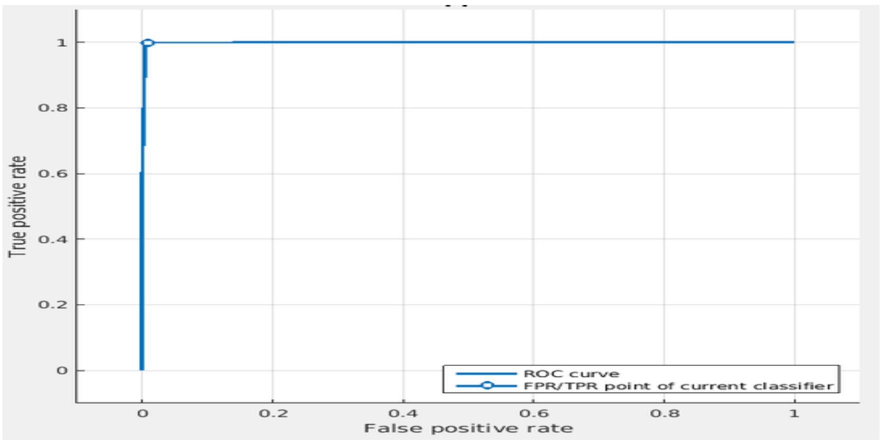


Fig. 4.25: ROC Curve for Quadratic SVM

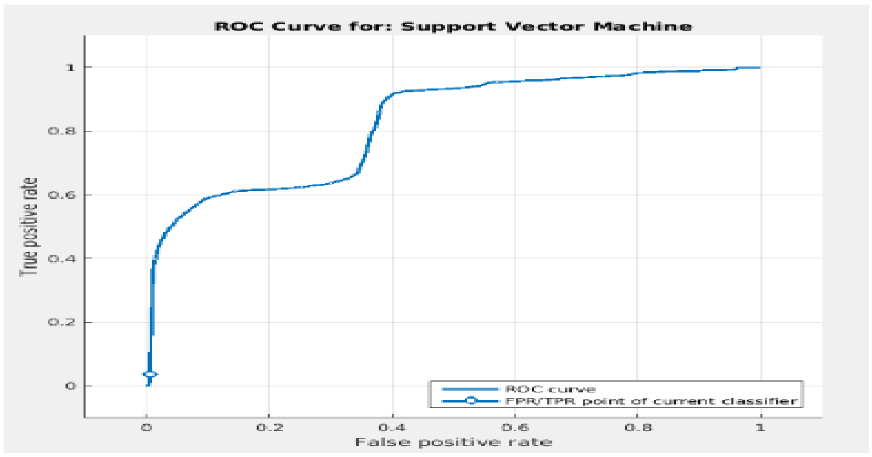


Fig. 4.26: ROC Curve for Cubic SVM

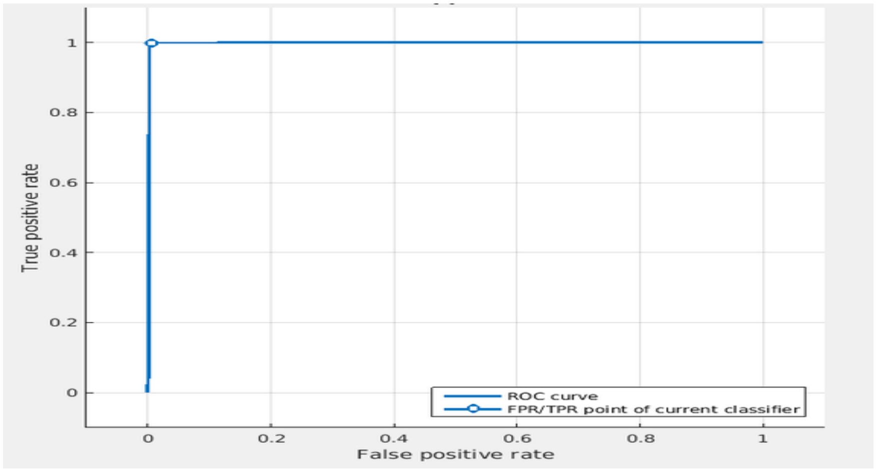


Fig. 4.27: ROC Curve for Fine Gaussian SVM

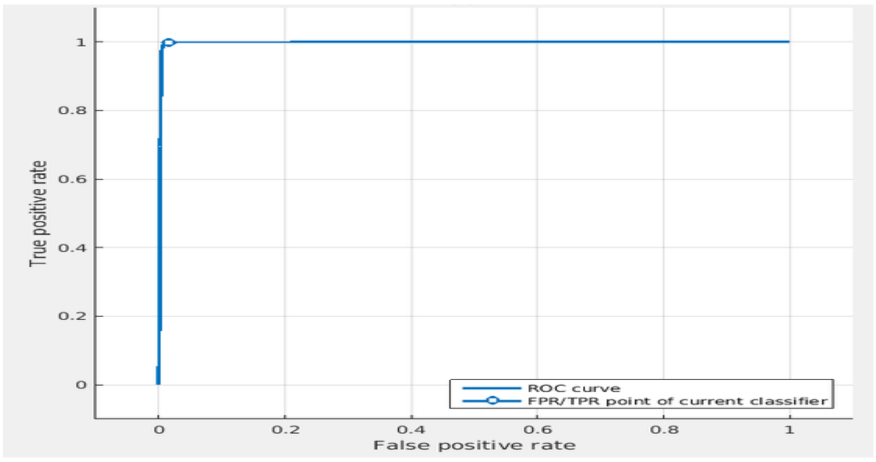


Fig. 4.28: ROC Curve for medium Gaussian SVM

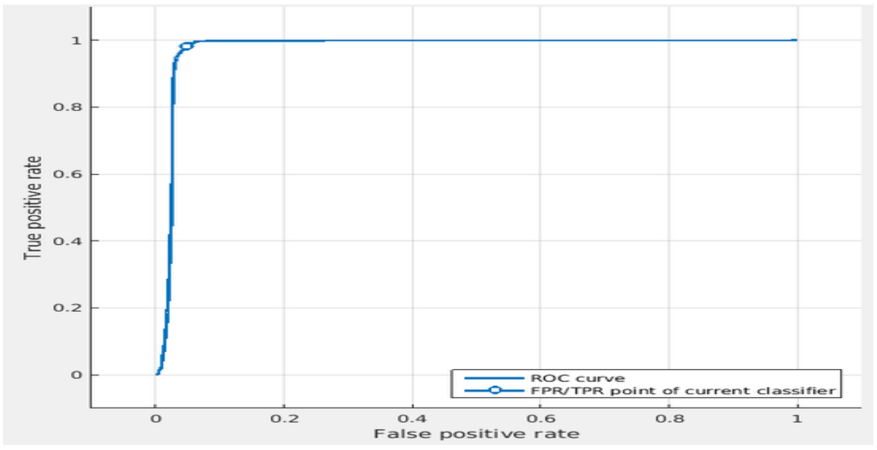


Fig. 4.29: ROC Curve for Coarse Gaussian SVM

4.10.3 Description of various Parameters after Training

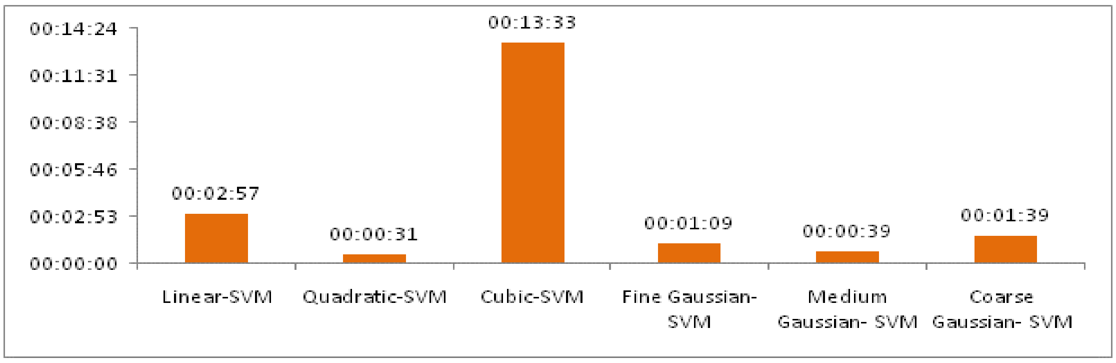
Different classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM have been used for the classification of the NSL-KDD dataset. Table- 4.10 shows the parameters of different classifiers after train the dataset.

Table- 4.10: Description of Parameters after Training

A	Linear SVM:	Training time: 00:02:57 Classifier options: type = SVM, kernel function: linear, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
B	Quadratic SVM:	Training time: 00:00:31 Classifier options: type = SVM, kernel function: Quadratic, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
C	Cubic SVM:	Training time: 00:13:33 Classifier options: type = SVM, kernel function: cubic, manual kernel scale = 1.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
D	Fine Gaussian SVM:	Training time: 00:01:09 Classifier options: type = SVM, kernel function: Gaussian, manual kernel scale = 1.6, kernel scale mode = manual, box

		constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
E	Medium Gaussian SVM:	Training time: 00:00:39 Classifier options: type = SVM, kernel function: Gaussian, manual kernel scale = 6.4, kernel scale mode = manual, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41
F	Coarse Gaussian SVM:	Training time: 00:01:39 Classifier options: type = SVM, kernel function: Gaussian, manual kernel scale = 26.0, kernel scale mode = auto, box constraint level = 1.0, multi class method = One-*vs*-One, standardize data = true Feature selection options: feature count before selection = 41, feature excluded = 0, feature included = 41

Further, the training time of the different linear and nonlinear kernel SVMs is depicted in Graph-4.6. Table-4.10 reflects the parameters and the training time of linear and nonlinear kernel SVMs after train the dataset. From this table, we deduced that Quadratic SVM is taking the optimal time i.e., 00:00:31 to train the dataset.



Graph-4.6: Training time of the linear and nonlinear kernel SVMs

4.11 Conclusion

In this chapter, the performance of various linear and non-linear Support Vector Machine classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM have been evaluated based on both KDD and NSL-KDD dataset with full 41-dimension features was used throughout the experiments and compared. To measure the performance of the different classifiers, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, Matthews’s Correlation Coefficient (MCC), and ROC Curve (AUC) were used.

The experimental results exposed that the classifier that performs well on the original KDD’99 dataset does not reveal the same result with the NSL-KDD dataset. This proves that the NSL-KDD dataset represents the more realistic result for evaluation of different linear and non-linear Support Vector Machine classifiers and compared to the KDD’99 dataset. Among various tested classification techniques, Quadratic SVM and Cubic SVM show the best result i.e. 100% each for the NSL-KDD dataset (Overall Accuracy) and ROC Curve (AUC) result reveals that Fine Gaussian SVM gives the best performance i.e. 99.98% as compared to other tested classifiers. According to time constraints, it is deduced that Quadratic SVM is taking the optimal time i.e., 00:00:31 to train both the dataset. The experimental results of the Support Vector Machine revealed that the NSL-KDD dataset shows a more realistic result as compared to the KDD’99 dataset.

CHAPTER-5

PERFORMANCE COMPARISON OF ARTIFICIAL NEURAL NETWORK (ANN) AND SUPPORT VECTOR MACHINE (SVM)

Chapter-5

Performance Comparison of Artificial Neural Network (ANN) and Support Vector Machine (SVM)

5.1 Introduction

Cybersecurity has become indispensable because of the prolific use of the Internet for a wide range of applications in multiple dimensions. This is coupled with the exponential growth of Internet users in all types of financial transactions like, banking, on-line purchase-including e-commerce, e-government, smart mobiles, e-mails, etc. which reached 3.39 billion globally (https://www.Statista.com/). Correspondingly, the cyber-attacks also equally gained momentum among the miscreants on the Internet domain which according to a timeline of notable targeted attack incidents during 2016 published in Symantec cybercrime report in 2017 shows that the cybercriminals got away with US$81 million from Bangladesh's central bank by exploiting weaknesses in the bank's security to infiltrate its network and steal its SWIFT credentials, allowing them to make the fraudulent transactions. Further, malicious emails were the instrument for cyber-attacks during 2016, used by everyone from state-sponsored cyberespionage groups to mass-mailing ransomware gangs. One in 131 emails sent were malicious which record the highest rate in five years.

Moreover, destructive malware was used in cyber-attacks against power stations in Ukraine (https://www.symantec.com/). Again, the Norton Cyber Security Insights Report, 2016 visualized that, total consumers globally affected by cybercrime in the past year reached 689.4 million (31%), the total financial cost of cybercrime in the past year touched to $125.9 billion (USD), most common cybercrimes affected to 18% for account password compromised, 16% for the hacking of emails, 15% theft of mobile devices and 22% for credit card fraud, followed by 21% for account password compromised and 19% for email hacking for US (https://www.symantec.com/).

Given the stupendous growth of internet users coupled with the transactions in multiple ways on the Internet domain, information security became indispensable to manage the issues smoothly and it is one of the major issues in the organizations that share a major chunk of their budgets in attempting to manage risk and alleviate intrusions (Caballero, 2009). Hence, security is indispensable in all organizations, enterprises to protect the application, confidential including the private data, patient privacy, financial information, safeguarding the trade secrets of the organization; company (Enterprise Applications Administration, 2014). Despite all security measures supported by the enactment of an astringent law against such cybercrime, there is a colossal growth of intrusion in every quarter. Therefore, two powerful classification algorithms i.e. Artificial Neural Network and Support Vector Machine are used to detect intrusion using KDD and NSL-KDD datasets.

5.2 Algorithms

For accomplishing the objectives of the study, the following neural network algorithms and some linear and kernel classification algorithms from the family of Support Vector Machine have been employed. In this study, the top 6 (six) best neural network classifiers and the top 6 (six) best set of supervised learning methods are selected and used for classification among various tested Support Vector Machine Algorithms based on evaluation performance. A set of neural network algorithms from various classification techniques consisting of Learning Vector Quantization (LVQ), Radial Basis Function Neural Network (RBFN), Decremental Radial Basis Function Neural Network (DECR-RBFN), Evolutionary Radial Basis Function Neural Network (EVRBFN), Multilayer Perceptron with Back-propagation Training (MLP-BP), Self-Optimizing Neural Networks (SONN) and some linear and kernel classification algorithms consisting of Linear SVM, Quadratic SVM, Cubic-SVM , Fine Gaussian SVM, Medium Gaussian SVM, Coarse Gaussian SVM algorithms are evaluated and compared the performance using both KDD (training & test) and NSL-KDD (training & test) dataset. The detailed experiments of all the neural network algorithms and support vector machine classifiers are described in Chapter-2, Chapter-3 and Chapter-4 respectively.

5.2.1 Artificial Neural Network (ANN)

The Artificial Neural Network i.e. ANNs symbolizes the notion of interconnecting of approximately 100 billion neurons i.e. nerve cells in the brain which communicate nerve signals to and from in the brain and it functions as a mathematical model that simulate the structure and functionalities of the biological neural network. The biological neurons comprise the following six components. (Bullinaria, 2015) as shown in Table-5.1.

Table-5.1: Description of six Biological Components

1	Neurons	Facilitate encoding their activations as a series of electrical pulses.
2	Neurons' Cell Body (Soma)	Processes incoming activations and converts them into output activations.
3	Neuron's nucleus	Includes the genetic material in the form of Deoxyribonucleic acid (DNA).
4	Dendrites	Fibers exhaled from the cell body provide accessible places to receive activation from other neurons.
5	Axons	Fibers perform as communication channels for sending activation to other neurons.
6	Synapses	Junctures allow signal transmission between the axons and dendrites.

The fundamental working phenomena of a biological neuron that after receiving inputs from other sources combine them and performs a nonlinear operation on the result, and then the result is presented. The relationships of four different biological components are shown in Fig.5.1.

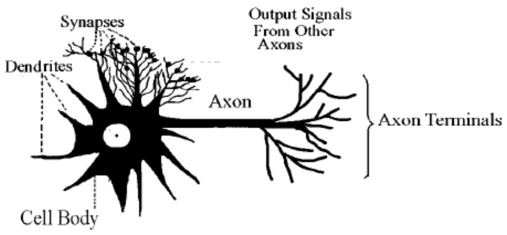


Fig. 5.1: Relationships of Four Biological Components
Source: Bangal, 2009

An artificial neuron is the constituent of every Artificial Neural Network and it is a mathematical model known as a function. Artificial Neural Network performs on three sets of rules such as (i) Multiplication, (ii) Summation, and (iii) Activation (Krenker *et al.*, 2011). Initially, each input in the artificial neuron is multiplied with the individual weight to obtain the weighted input values which along with the bias are summed up through summation function and finally the weighted sum of earlier weighted inputs including bias pass through activation function which is also known as transfer function (Farhat and Roohi, 2013). This phenomenon is explained in Figure- 5.2.

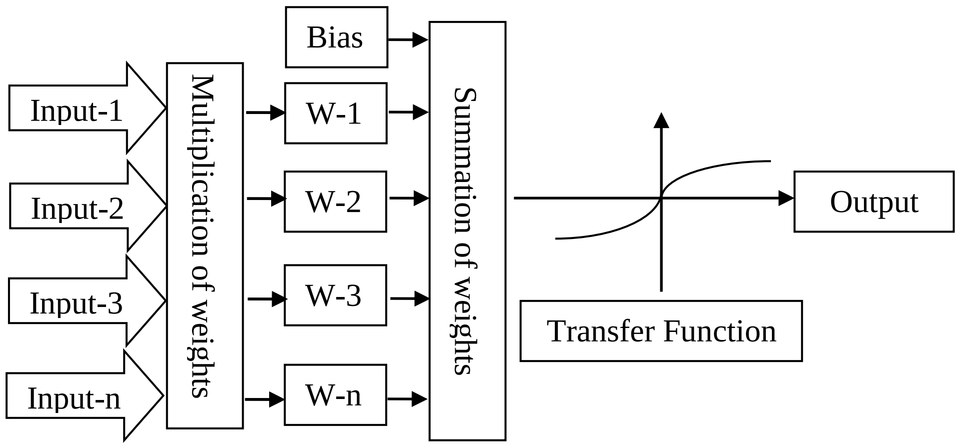


Fig. 5.2: Working Model of Neuron

Artificial Neural Network is relatively different from the algorithmic program as it is involved in the process of producing knowledge of a given data or new set of data which absent during the learning process. In the learning process, the neural network uses statistical methods to optimize a set of internal weights that correspond to each of the inputs to each of the probable outputs (Patnaik, 2001). Artificial Neural Network operates on different approaches forms of expert systems that rely on the ability of an expert to capture knowledge in the form of a knowledge representation such as rules. On contrary, Artificial Neural Network is built assuming that no such explicit knowledge exists rather, it uses mathematical techniques to learn the recognition technique of a particular type of pattern by combining the different features from a set of training examples (Patnaik, 2001). Artificial Neural Network is limited to overfitting, fixed topology, and slow convergence (Tang and Ghorbani, 2003).

All the Neural Network classifiers i.e. Learning Vector Quantization (LVQ), Radial Basis Function Network (RBFN), Decremental Radial Basis Function Network (DECR-RBFN), Evolutionary Radial Basis Function Network (EV-RBFN), Multilayer Perceptron Back Propagation (MLP-BP), and Self-Organising Neural Networks (SONN)were used for the experiments. Among all these classifiers, the SONN algorithm has performed the best result for both KDD and NSL-KDD Datasets. The detailed experiments have been performed in Chapter-2 and Chapter-3 of the work.

5.2.2 Support Vector Machine (SVM)

Support Vector Machine (SVM) relates to a supervised machine learning algorithm that is used for both classification and regression. This otherwise can be explained that on a given labeled training data i.e. supervised learning, the algorithm results in an optimal hyperplane which in two-dimensional space is a line that divides a plane into two parts wherein each class lay on either side (Aylien, 2016). It is employed mainly for the classification of the text that includes category assignment, spam detection, and sentiment analysis including image recognition challenges, performing particularly well in aspect-based recognition and color-based classification. It is also employed for handwritten digit recognition, such as postal automation services. (Aylien, 2016). Statistical learning techniques based on risk minimization such as Support Vector Machine (SVM) are the dominating classification schemes. Compared to Artificial Neural Network, Support Vector Machine has the positive implication that includes (i) Structural Risk Minimization techniques which minimize a risk upper bound on the VC-dimension, (ii) Maximising the separation margin between the classes in hyperplane compared to other hyperplanes used for separating the data and (3) the power of SVM lies in using a kernel function to transform data from the low dimension space to the high dimension space and construct a linear binary classifier (Tang and Ghorbani, 2003).

The recent studies visualised that, the use of Support Vector Machine is more prevalent in Intrusion Detection System studies due to its accurate classification result, robust to noise, less overfitting. Support Vector Machine is not only a useful tool for insolvency analysis, in the case of non-regularity in the data but also deliver a unique solution, since the optimality problem is convex and this is an advantage compared to Neural Networks, which have multiple solutions associated with local minima and for this reason may not be robust over different samples (Auria and Moro, 2008).

All the classifiers of Support Vector Machine such as Linear SVM, Quadratic SVM, Cubic-SVM , Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM have been used for the experiments. Among these classifiers, Quadratic SVM and Cubic-SVM classifier has performed the best result for both KDD and NSL-KDD Datasets. The detailed experiments have been performed in Chapter-4 of the work.

5.3 Justification for Comparison between Artificial Neural Network and Support Vector Machine in Intrusion Detection System using KDD and NSL-KDD Dataset

Information is one of the indispensable assets of an organization where a good quantum of information is stored and processed in network-based computers. The confidentiality, integrity, and availability of the system resources have raised the vulnerability of these systems to security threats, attacks, and intrusions. Many authors, scientists, researchers have used different mechanisms to detect intrusion in the network. Some key points as discussed below were observed while reviewing the literature to justify the research problem.

Kaur *et al.* (2015), observed that feature selection is significant in Intrusion Detection and according to them, its relevance counts as the basic measurement in feature selection techniques. They applied various feature selection techniques to the NSL-KDD dataset for reduced training & test data sets. They also used Naive Bayes Classifier to classify the data set and compared all the experimented results by using different performance metrics like TP rate, FP rate, Precision, ROC area, Kappa Statistic, and Classification Accuracy.

Garg and Khurana (2014) observed that network security mechanisms require more attention to improve speed and precision due to the growth of network-based applications. They also found that Intrusion detection systems (IDSs) are used to detect intrusive activities on the network due to evolving of new intrusion types which caused a serious threat to network security. The authors also concluded that machine learning and classification algorithms help to design "Intrusion Detection Models" which can classify the network traffic into intrusive or normal traffic. The authors presented a comparative performance of NSL-KDD based dataset compatible classification algorithms. They evaluated these classifiers in WEKA (Waikato Environment for Knowledge Analysis) environment using 41 attributes. They included 94,000 instances from the complete KDD'99 dataset in the training dataset and over 48,000 instances in the testing data set. They applied Garrett's Ranking Technique to rank different classifiers according to their performance. The Rotation Forest classification approach outperformed the rest. Devaraju and Ramakrishnan (2014) in their study viewed that, Intrusion Detection Systems are a challenging task for finding the user as a normal user or attack user in any organizational information systems or IT Industry. The Intrusion Detection System is an effective method to deal with the kinds of problems in networks. Different classifiers are used to detect the different kinds of attacks in networks. In the paper, the performance of intrusion detection is compared with various neural network classifiers. In the research, the four types of classifiers are used such as Feed Forward Neural Network (FFNN), Generalized Regression Neural Network (GRNN), Probabilistic Neural Network (PNN), and Radial Basis Function Neural Network (RBFNN). The performance of the full-featured KDD Cup 1999 dataset is compared with that of the reduced featured KDD Cup 1999 dataset. The MATLAB software is used to train and test the dataset and the efficiency and False Alarm Rate are measured. It is proved that the reduced dataset is performing better than the full-featured dataset.

A neural network algorithm is used to detect intrusions in the network. The neural network algorithms are popular for their ability to 'learn' the patterns in a given environment and thus, can be trained to detect intrusions by recognizing patterns of an intrusion. In this work, the author performed a comparative study of Multilayer Feed Forward, Elman Back Propagation (EBP), Cascaded Forward Back Propagation (CFBP), and Self-Organizing Feature Map (SOFM) neural network-based intrusion detection systems. In this study, the author worked on the well-structured KDD CUP'99 dataset (Nazir, 2013).

Ibrahim *et al.* (2013), applied SOM on KDD'99 and NSL-KDD datasets and their experiment visualize the better result of binary classification on the KDD'99 dataset than that of the NSL dataset. Bhoria and Garg (2013) statistically analyzed an NSL-KDD dataset with a 6-fold cross-validation technique. The analyses concluded that when a more relevant feature set is applied for classification then it improves the performance of the Intrusion Detection System. This improved performance is achieved due to increased classification accuracy and reduced classification time. This increased classification accuracy provides a much better detection capacity of IDS towards attacks and hence results in safe networking to users. Bajaj and Arora (2013) found that the Simple Cart algorithm performs better on Intrusions that are hard to detect on the system. Giray and Polat (2013) visualize that, data mining provides a functional environment and set of tools for processing large datasets i.e. Intrusion Detection Systems (IDS) logs. Researchers improve existing IDS models by comparing the performance of different algorithms on these same datasets. It is very essential to keep in mind that an Intrusion Detection System often has to work in a noisy network environment. Network noise is one of the most challenging issues for efficient threat detection and classification. In this study, normal and noisy datasets for the network IDS domain are used and various classification algorithms are evaluated. The results show that an evaluation of algorithms without noise is misleading for Intrusion Detection Systems since algorithms that perform best without noise do not necessarily achieve the same in a really noisy environment. Moreover, the refined NSL-KDD dataset allows a more realistic evaluation of various algorithms than the original KDD'99 dataset.

Shah and Trivedi (2012) mentioned that detecting unknown or modified attacks is one of the recent challenges in the field of IDS. Anomaly-based IDS can play a very important role in this case. In the paper, they focussed on ANN which is used to address these issues. Several researchers have already shown the importance of the various Artificial Neural Network (ANN) based techniques for anomaly detection. In this paper, the authors also discussed the Simple and Hybrid ANN-based approach for anomaly detection. In a simple approach, they discussed on Back

Propagation Neural Network (BPNN), Self-Organizing Maps (SOM), Support Vector Machine (SVM), and Simulated Annealing Neural Network (SA) which are used for anomaly detection. In a hybrid approach, they focussed on how more than one above technique can be used. The authors compared the different ANN-based techniques in terms of training time, the number of epochs required, converge rate, detection rate, learning approach, etc. The authors viewed that, to avail the advantages of more than one ANN technique, researchers are using a combination of the more than one technique (multi-layer approach).

Imran *et al.* (2012), applied features transformation and optimum subset selection using Linear Discriminate Analysis (LDA) algorithm and Genetic Algorithm (GA) respectively. Radial Basis Function (RBF) is adapted as a features classifier. They applied cross-validation on 20% of the NSL KDD training dataset for training and testing. Empirical results show that the new proposed system gives a better and robust representation of an ideal intrusion detection system while having a reduced number of features, low false alarms, high detection rate, and minimum computation cost.

Mulay *et al.* (2010), studied Intrusion Detection Systems using a Support Vector Machine (SVM) and decision tree. They concluded that c1assification applications can solve multi-class problems. Decision-tree-based Support Vector Machine which combines Support Vector Machine and decision tree can be an effective way of solving multi-class problems. They viewed that; this method can decrease the training and testing time, increasing the efficiency of the system. The construction order of the binary tree has a great influence on the classification performance. In the paper, they studied an algorithm; Tree-structured multiclass SVM, which was used for classifying data. Sadoddin and Ghorbani (2007) conducted blind experiments on unsupervised techniques on the KDD’99 dataset to analyze the performance of unsupervised techniques considering their main design choice. In this paper algorithms of the three categories are studied as Clustering techniques, Unsupervised Support Vector Machine, and K Nearest-Neighbour.

Gharibian and Ghorbani (2007) pointed out that, an intrusion detection is an effective approach for dealing with various problems in the area of network security. In the paper, the authors presented a comparative study of using supervised probabilistic and predictive machine learning techniques for intrusion detection. Two probabilistic techniques Naive Bayes and Gaussian and two predictive techniques decision tree and random forests were employed in their study. Different training datasets constructed from the KDD'99 dataset were also employed for training. The ability of each technique for detecting four attack categories (DoS, Probe, R2L, U2R) has been compared. The statistical results of the sensitivity of each technique to the population of attacks in a dataset have also been reported. The authors compared the performance of the techniques and also investigated the robustness of each technique by calculating their standard deviations concerning the detection rate of each attack category.

Pervez *et al*. (2006), discussed that Intrusion Detection is a major focus of research in the security of computer systems and networks. Their paper presented an analysis of Artificial Neural Networks (ANN) which is being used in the development of effective Intrusion Detection Systems for computer systems and computer networks. The ANNs technologies, which are discussed, are designed to detect instances of the access of computer systems by unauthorized individuals and the misuse of system resources. A review of the foundations of Intrusion Detection Systems and other Artificial Neural Networks, which are the focus of current development efforts, is also included in the paper. The paper also included the results of comparative analysis of different Artificial Neural Networks in Intrusion Detection. They viewed that, a discussion of the future Artificial Neural Network technologies guarantees to enhance the ability of computer systems to detect intrusions. Novikov (2005) in the thesis evaluated the results of the average performance of each system regarding KDD Cups Winner, Runner UP, and Multi-Classifier Model based on their performance in classifying the attacks into 4 different groups. MLP and RBF IDS were averaged based on their performance in classifying the attacks and detecting and classifying the unknown attack.

From the above study, it was found that no comparative study has been performed using the two different types of classifiers ANN (SONN) and SVM (Cubic and Quadratic) for both KDD and NSL-KDD dataset. We have chosen these classifiers as it shows the best results among other classifiers. The detailed experiments are performed in Chapter-2, Chapter-3 and Chapter-4 respectively. However, the performance comparison between Neural Network and Support Vector Machine using KDD and NSL-KDD datasets is depicted below.

5.4 Dataset Description and Performance Evaluation Measures

To discuss a brief description of the dataset, the Lincoln Laboratory of Massachusetts of Technology, a private research university in Cambridge is the pioneer institute that developed DARPA (Defence Advanced Research Projects Agency) Datasets in 1998, 1999, and 2000. The data sets of 1998 & 1999 are the result of the DARPA Intrusion Detection Evaluation while datasets of 2000 concentrate on Intrusion Detection Scenario-Specific (http://www.Ll.mit.edu/ ideval/data). Tavallaee *et al.* (2009), visualized that, the data captured in DARPA’98 Intrusion Detection System evaluation comprises 7 weeks of network traffic data (5 weeks for training purpose and 2 weeks for testing purpose) which can be processed into 5 million connection records each with 100 bytes. It may be mentioned that two weeks of test data constitute approximately 2 million connection records. The KDD’99 dataset which originally acclaims from the DARPA’98 dataset comprises around 4,900,000 single connection vectors where every 41 features constitute and labeled as normal or an attack with one specific attack type. The authors further proposed that NSL-KDD dataset (http://nsl.cs.unb.ca/NSL-KDD/) is a refined and condensed dataset of the original KDD’99 dataset that constitutes the same 41 features and one class attribute. It is composed of 21 classes and is covered under four classes of attacks as Probe attacks, User to Root (U2R) attacks, Remote to Local (R2L) attacks, and Denial of Service (DoS) attacks. The attack types correspond to the attack name in the NSL-KDD dataset.

In this study, the multi-class NSL-KDD dataset is converted to the binary class dataset by combining different types of anomalies. So, now we have used a binary classification strategy. i.e. 2-class (Normal and Anomaly). The performance of the classifiers is monitored and measured by using different performance matrices such as,

(i)	**Overall Accuracy**	:	For the performance analysis, the performance metric (accuracy) is used with the help of the confusion matrix. The accuracy of a test is its capability to distinguish the positive and negative cases accurately. To estimate the accuracy of a test, we should calculate the proportion of true positive and true negative in all evaluated cases.
(ii)	S**pecificity**	:	The specificity is the proportion of the TN and (TN+FP) and with the higher specificity fewer positive cases are labeled as negatives, so this ratio can be regarded as the percentage of negative cases correctly classified as belonging to the negative class. The specificity of a test is its ability to determine the negative cases correctly. To estimate it, we should calculate the proportion of true negative in negative cases.
(iii)	**Sensitivity**	:	The proportion of cases that are TP for all the cases that are positive in the diagnostic test (TP+FN) is called sensitivity. The sensitivity of a test is its ability to conclude the positive cases correctly. To estimate it, we should compute the proportion of true positive in positive cases.
(iv)	**G-Mean**	:	The Geometric Mean (G-Mean) or G-measure is another metric used to evaluate the performance results by using both specificity and sensitivity. It ranges from 0 to 1 and an attribute that is perfectly correlated to the class provides a value of 1.

(v) **Precision (PPV)**: Precision is also called a positive predictive value that measures the exactness or quality.

(vi) **Recall** : Recall is also known as sensitivity which measures completeness or quantity.

(vii) **F-Measure** : F- Measure or balanced F-score combines precision and recall is the harmonic mean of precision and recall.

(viii) **Matthews's Correlation Coefficient (MCC)**:

Matthews's correlation coefficient (MCC) (Boughorbel *et al.*, 2017) measures the quality of two-class binary classification by biochemist Brian W. Matthews in 1975. It returns a value between -1 and +1.

5.5 Performance Comparison Framework

The performance comparison framework consists of two different data mining techniques such as Artificial Neural Network and Support Vector Machine. This is intended to make a comparison out of the best performers from both Artificial Neural Network and Support Vector Machine. The detailed experiments using different Artificial Neural Network algorithms are discussed in Chapter-2 and Chapter-3 and experiments using various Support Vector Machine classifiers are discussed in Chapter-4 of the work. From Artificial Neural Network experiments, SONN performed the best among all other neural network algorithms, and from Support Vector Machine experiments, both Quadratic and Cubic performed the best result. Finally, the best performing algorithms were compared individually using both KDD and NSL-KDD dataset. Accordingly, the performance comparison framework has been designed and placed in Fig. 5.3.

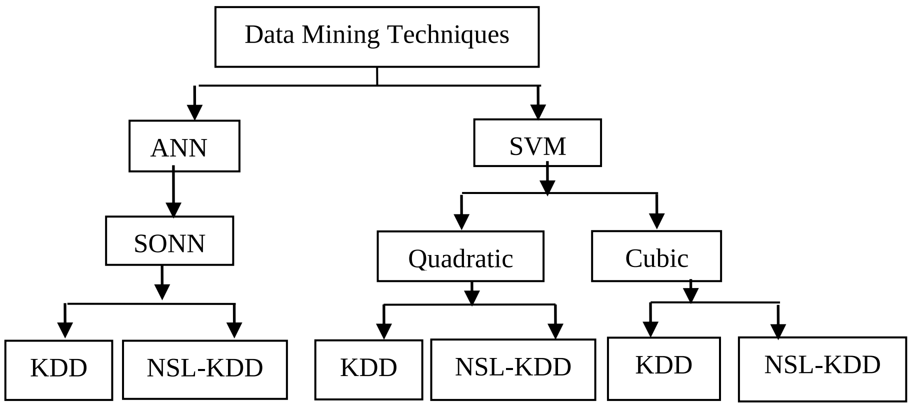


Fig.5.3: Performance Comparison Framework

5.5.1 Experimental Setup

The whole neural network experiments are done by using Knowledge Extraction based on Evolutionary Learning (KEEL), which is an open-source (GPLv3) Java software available at (http://sci2s.ugr.es /keel/ datasets.php) used for a large number of different knowledge data discovery tasks (http://www.keel.es/) and Support Vector Machine experiments are performed using Matrix Laboratory (MATLAB) (Bryant and Garber, 1999 & Sumathi and Paneerselvam, 2010) software. We are using Windows 7 Home Basic (64-bit) as the testbed operating system, Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz processor, 3GB of RAM. To test different classification algorithms of Artificial Neural Network and Support Vector Machine, both original KDD'99 and NSL-KDD Dataset which is an enhanced edition of original KDD'99 dataset is used for the whole experiments consisting of 41 features and 10-fold cross-validation technique is applied for comparison between Artificial Neural Network and Support Vector Machine to know the best performance for Intrusion Detection System.

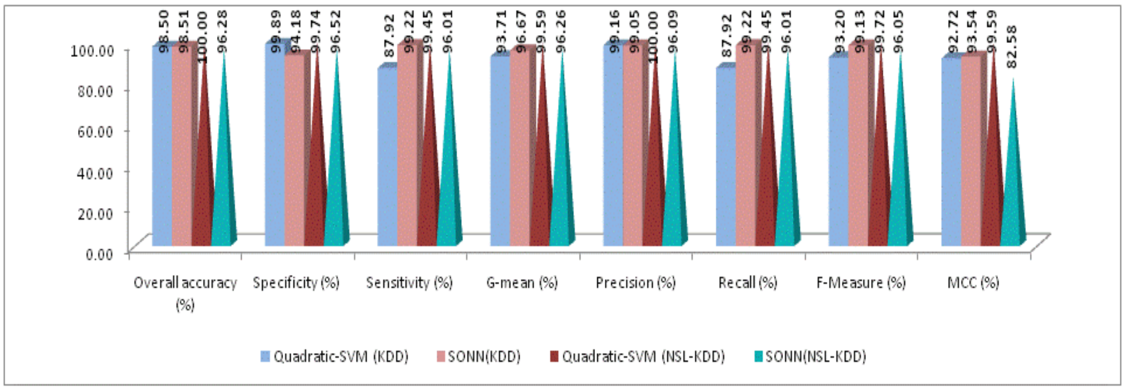
All the eight performance matrices have been grouped into two sets where Set-1 comprises, i) Overall Accuracy , ii) Specificity, iii) Sensitivity and iv) G-mean and Set-2 constitute i) Precision, ii) Recall, iii) F-Measure and iv) Matthews's Correlation Coefficient (MCC). These two sets of performance matrices are tested individually using both KDD and NSL-KDD dataset for both Neural Network and Support Vector Machine. Further, to find out the best performance for Intrusion Detection System between the two algorithms comprehensive comparison was performed for (i) Quadratic-SVM and SONN for both KDD and NSL-KDD dataset and (ii) Cubic-SVM and SONN for both KDD and NSL-KDD dataset.

5.6 Comprehensive Comparison between Quadratic-SVM and SONN for both KDD and NSL-KDD (training) dataset

The comprehensive comparison of Quadratic-SVM (KDD) & SONN (KDD) and Quadratic-SVM (NSL-KDD) & SONN (NSL-KDD) training dataset is placed in Table-5.2 by using different performance matrices i.e., Overall Accuracy , Specificity, Sensitivity and G-Mean, Precision, Recall, F-Measure and MCC with the corresponding Graph- 5.1.

Table-5.2: Performance comparison of Quadratic-SVM & SONN for both KDD and NSL-KDD (training) dataset

Matrices	**Methods**			
	Quadratic-SVM (KDD)	**SONN (KDD)**	**Quadratic-SVM (NSL-KDD)**	**SONN (NSL-KDD)**
Set-1				
Overall Accuracy (%)	98.50	98.51	100	96.28
Specificity (%)	99.89	94.18	99.74	96.52
Sensitivity (%)	87.92	99.22	99.45	96.01
G-mean (%)	93.71	96.67	99.59	96.26
Set-2				
Precision (%)	99.16	99.05	100	96.09
Recall (%)	87.92	99.22	99.45	96.01
F-Measure (%)	93.20	99.13	99.72	96.05
MCC (%)	92.72	93.54	99.59	82.58



Graph- 5.1: Performance comparison of Quadratic-SVM & SONN for both KDD and NSL-KDD (training) dataset

5.6.1 Results and Analysis

The performance comparison of Quadratic-SVM and SONN for both KDD and NSL-KDD training dataset placed in Table-5.2 found that, in Set-1, the Overall Accuracy matrix for SONN algorithm using KDD dataset is higher i.e. 98.51% compared to Quadratic-SVM which comes to 98.50%. But, the Overall Accuracy matrix of Quadratic- SVM using the NSL-KDD dataset is higher i.e. 100% compared to the SONN algorithm which comes to 96.28%. The Specificity matrix for Quadratic-SVM using the KDD dataset is higher i.e. 99.89% compared to SONN algorithm i.e. 94.18%.

Again, the Quadratic-SVM using NSL-KDD dataset is higher i.e.99.74% compared to SONN algorithm i.e. 96.52%. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher i.e. 99.22% compared to Quadratic-SVM which comes to 87.92%. But, Quadratic-SVM using NSL-KDD dataset is higher i.e. 99.45% compared to SONN i.e. 96.01%. For G-mean matrix, SONN algorithm using KDD dataset is higher i.e. 96.67% compared to Quadratic-SVM which comes to 93.71%. But, Quadratic-SVM shows high i.e. 99.59% in NSL-KDD dataset compared to SONN i.e. 96.26%.

In Set-2, for Precision matrix using KDD dataset, Quadratic- SVM is higher i.e. 99.16% while in SONN algorithm it comes to 99.05% but, for the NSL-KDD dataset, Quadratic- SVM is higher i.e.100% while in SONN algorithm it comes to 96.09%. In Recall matrix, SONN algorithm using KDD dataset comes to 99.22%

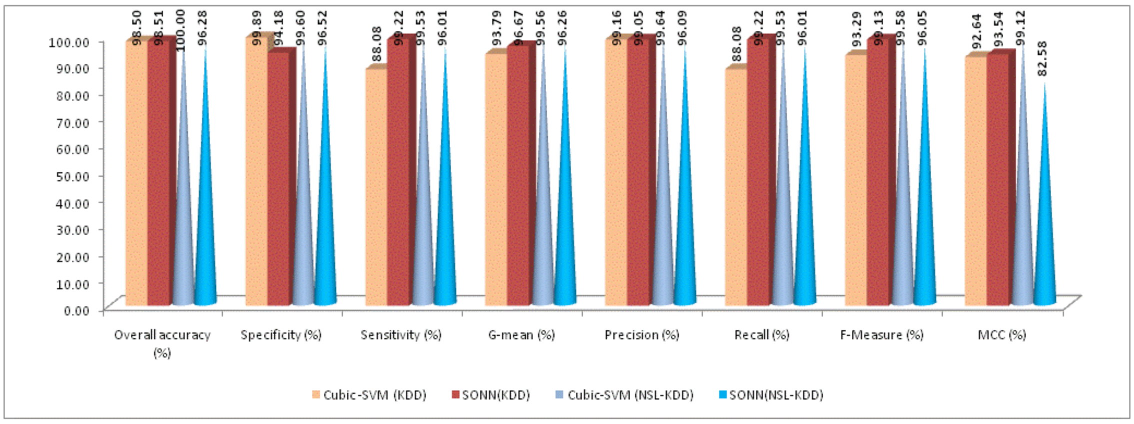
which is higher as compared to Quadratic- SVM which comes to 87.92%. But, using NSL-KDD dataset, Quadratic- SVM is higher i.e. 99.45% as compared to 96.01% in SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher i.e. 99.13% as compared to Quadratic-SVM which comes to 93.20% but, Quadratic-SVM using NSL-KDD dataset is higher i.e. 99.72% as compared to SONN algorithm i.e. 96.05%. For MCC, SONN algorithm using KDD dataset is higher i.e. 93.54% as compared to Quadratic-SVM which comes to 92.72% but, for the NSL-KDD dataset, Quadratic- SVM is higher i.e. 99.59% than SONN algorithm i.e. 82.58%.

5.7 Comprehensive Comparison between Cubic-SVM and SONN for both KDD and NSL-KDD (training) dataset

The comprehensive comparison of Cubic-SVM (KDD) & SONN (KDD) and Cubic-SVM (NSL-KDD) & SONN (NSL-KDD) training dataset is shown in Table-5.3 by using different performance matrices i.e. Overall Accuracy, Specificity, Sensitivity and G-Mean, Precision, Recall, F-Measure and MCC with the corresponding Graph- 5.2.

Table-5.3: Performance comparison of Cubic-SVM and SONN for both KDD and NSL-KDD (training) dataset

Matrices	**Methods**			
	Cubic -SVM (KDD)	**SONN (KDD)**	**Cubic-SVM (NSL-KDD)**	**SONN (NSL-KDD)**
Set-1				
Overall Accuracy (%)	98.50	98.51	100	96.28
Specificity (%)	99.89	94.18	99.60	96.52
Sensitivity (%)	88.08	99.22	99.53	96.01
G-mean (%)	93.79	96.67	99.56	96.26
Set-2				
Precision (%)	99.16	99.05	99.64	96.09
Recall (%)	88.08	99.22	99.53	96.01
F-Measure (%)	93.29	99.13	99.58	96.05
MCC (%)	92.64	93.54	99.12	82.58



Graph- 5.2: Performance comparison of Cubic-SVM and SONN for both KDD and NSL-KDD (training) dataset

5.7.1 Results and Analysis

The performance comparison of Cubic-SVM and SONN for both KDD and NSL-KDD dataset placed in Table-5.3 found that, in Set-1, the Overall Accuracy matrix for SONN algorithm using KDD dataset is higher i.e. 98.51% compared to Cubic- SVM which comes to 98.50%. But the Cubic-SVM using NSL-KDD dataset is higher i.e. 100% compared to SONN algorithm which comes to 96.28%. In Specificity matrix for Cubic-SVM using KDD dataset is higher i.e. 99.89% compared to SONN algorithm i.e. 94.18%. Again, the Cubic-SVM using NSL-KDD dataset is higher i.e.99.60% compared to SONN i.e. 96.52%. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher i.e. 99.22% compared to Cubic-SVM which comes to 88.08%. But, Cubic-SVM using NSL-KDD dataset is higher i.e. 99.53% compared to SONN i.e. 96.01%. For G-mean matrix, SONN algorithm using KDD dataset is higher i.e. 96.67% compared to Cubic-SVM which comes to 93.79%. But, in NSL-KDD dataset, Cubic-SVM figures high i.e. 99.56% compared to SONN i.e. 96.26%.

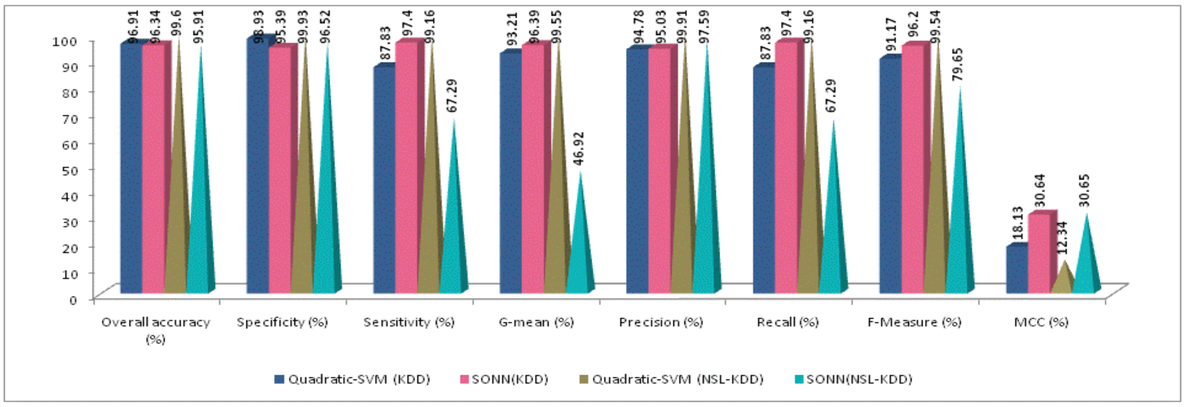
In Set-2, for Precision matrix using KDD dataset, Cubic-SVM is higher i.e. 99.16% while in SONN algorithm it comes to 99.05% but, for the NSL-KDD dataset, Cubic-SVM is higher i.e.99.64% while in SONN algorithm it comes to 96.09%. In Recall matrix, SONN algorithm using KDD dataset comes to 99.22% which is higher as compared to Cubic-SVM which comes to 88.08%. But, using NSL-KDD dataset, Cubic-SVM is higher i.e. 99.53% as compared to 96.01% in SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher i.e. 99.13% as compared to Cubic-SVM which comes to 93.29% but, Cubic-SVM using NSL-KDD dataset is higher i.e. 99.58% as compared to SONN algorithm i.e. 96.05%. For MCC, SONN algorithm using KDD dataset is higher i.e. 93.54% as compared to Cubic-SVM which comes to 92.64% but, for the NSL-KDD dataset, Cubic-SVM is higher i.e. 99.12% than SONN algorithm i.e. 82.58%.

5.8 Comprehensive Comparison between Quadratic-SVM and SONN for both KDD and NSL-KDD (test) dataset

The comprehensive comparison of Quadratic-SVM (KDD) & SONN (KDD) and Quadratic-SVM (NSL-KDD) & SONN (NSL-KDD) test dataset is placed in Table-5.4 by using different performance matrices i.e., Overall Accuracy, Specificity, Sensitivity and G-Mean, Precision, Recall, F-Measure and MCC with the corresponding Graph- 5.3.

Table-5.4: Performance comparison of Quadratic- SVM & SONN for both KDD and NSL-KDD (test) dataset

Matrices	**Methods**			
	Quadratic-SVM (KDD)	**SONN (KDD)**	**Quadratic-SVM (NSL-KDD)**	**SONN (NSL-KDD)**
Set-1				
Overall Accuracy (%)	96.91	96.34	99.60	95.91
Specificity (%)	98.93	95.39	99.93	96.52
Sensitivity (%)	87.83	97.40	99.16	67.29
G-mean (%)	93.21	96.39	99.55	46.92
Set-2				
Precision (%)	94.78	95.03	99.91	97.59
Recall (%)	87.83	97.40	99.16	67.29
F-Measure (%)	91.17	96.20	99.54	79.65
MCC (%)	18.13	30.64	12.34	30.65



Graph- 5.3: Performance comparison of Quadratic- SVM & SONN for both KDD and NSL-KDD (test) dataset

5.8.1 Results and Analysis

The performance comparison of Quadratic SVM and SONN for both KDD and NSL-KDD dataset placed in Table-5.4 found that, in Set-1, the Overall Accuracy matrix for Quadratic SVM classifier using KDD dataset is higher i.e. 96.91% compared to SONN which comes to 96.34%. But, again the Quadratic SVM using NSL-KDD dataset is higher i.e.99.60% compared to SONN algorithm which comes to 95.91%. In Specificity matrix for Quadratic SVM using KDD dataset is higher i.e. 98.93% compared to SONN algorithm i.e. 95.39%. Again, the Quadratic SVM using NSL-KDD dataset is higher i.e. 99.93% compared to SONN i.e. 96.52%. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher i.e. 97.40% compared to Quadratic SVM which comes to 87.83%. But, Quadratic SVM using NSL-KDD dataset is higher i.e. 99.16% compared to SONN i.e. 67.29%. For G-mean matrix, SONN algorithm using KDD dataset is higher i.e. 96.39% compared to Quadratic SVM which comes to 93.21%. But in the NSL-KDD dataset, Quadratic SVM figures high i.e. 99.55% compared to SONN i.e. 46.92%.

In Set-2, for Precision matrix using KDD dataset, SONN algorithm is higher i.e. 95.03% while in Quadratic SVM classifier comes to 94.78% but, for the NSL-KDD dataset, Quadratic SVM is higher i.e. 99.91% while in SONN algorithm comes to 97.59%. In Recall matrix, SONN algorithm using KDD dataset comes to 97.40% which is higher as compared to Quadratic SVM which comes to 87.83%. But, using

NSL-KDD dataset, Quadratic SVM is higher i.e. 99.16% as compared to 67.29% in SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher i.e. 96.20% as compared to Quadratic SVM which comes to 91.17% but, Quadratic SVM using NSL-KDD dataset is higher i.e. 99.54% as compared to SONN algorithm i.e. 79.65%. For MCC, SONN algorithm using KDD dataset is higher i.e. 30.64% as compared to Quadratic SVM which comes to 18.13% but, for the NSL-KDD dataset, again SONN algorithm is higher i.e. 30.65% than Quadratic SVM i.e. 12.34%.

5.9 Experimental Results/Findings

5.9.1 It could be observed from Table-5.2 that, SONN algorithm performs the best result using the KDD training dataset for Overall Accuracy, Sensitivity and G-mean which come to i.e. 98.51%, 99.22%, and 96.67% respectively. For Specificity, Quadratic-SVM is found to be the best i.e. 99.89%.

5.9.2 For Precision, Quadratic-SVM performs better than SONN i.e. 99.16%. SONN performed the best result using KDD dataset for Recall, F-Measure, and MCC, i.e. 99.22%, 99.13%, and 93.54% respectively.

5.9.3 It was found that Quadratic-SVM performed the best for Overall Accuracy i.e. 100% while using NSL-KDD training dataset as compared to SONN algorithm. Again, Quadratic-SVM performed the best result for other performance matrices, such as Specificity (99.74%), Sensitivity (99.45%), and G-mean (99.59%).

5.9.4 Again, Quadratic-SVM performed the best result in using NSL-KDD dataset for the performance matrices i.e. Precision (100%), Recall (99.45%), F-Measure (99.72%), and MCC (99.59%).

5.9.5 It could be observed from Tabel-5.3 that, SONN algorithm performs the best result using the KDD training dataset for Overall Accuracy (98.51%), Sensitivity (99.22%), and G-Mean (96.67%) as compared to Cubic- SVM. Further, Cubic-SVM performed the best result for Specificity (99.89%) and Precision (99.16%).

5.9.6 Finally it could be deduced that SONN shows the best performance for Recall (99.22%), F-Measure (99.13%) and MCC (93.54%).

5.9.7 On using of NSL-KDD training dataset and compared to SONN algorithm, Cubic-SVM shows the best result for all the performance matrices i.e. Overall Accuracy (100%), Specificity (99.60%), Sensitivity (99.53%), G-Mean (99.56%), Precision (99.64%), Recall (99.53%), F-Measure (99.58%) and MCC (99.12%).

5.9.8 From Table-5.4 it could be concluded that for both KDD and NSL-KDD test dataset again Quadratic-SVM is found to be the best among all other Support Vector Machine classifiers and SONN algorithm is found to be the best among all other Artificial Neural Network algorithms.

5.9.9 After using of KDD (test) dataset and compared to SONN algorithm, Quadratic-SVM is performing the best result for Overall Accuracy (96.91%) and Specificity (98.93%). But for Sensitivity and G-Mean, SONN algorithm performs the best result i.e. 97.40% and 96.39% respectively.

5.9.10 Again for Precision, Recall, F-Measure, and MCC, SONN algorithm performs the best result i.e. 95.03%, 97.40%, 96.20%, and 30.64% respectively.

5.9.11 After using of NSL-KDD (test) dataset and compared to SONN algorithm, Quadratic-SVM is performing best result for all the performance matrices i.e. Overall Accuracy (99.60%), Specificity (99.93%), Sensitivity (99.16%), G-Mean (99.55%), Precision (99.91%), Recall (99.16%) and F-Measure (99. 54%). But for MCC, SONN algorithm is performing the best result i.e. 30.65%.

The result from Table-5.2 deduced that, in Set-1, the Overall Accuracy matrix for SONN algorithm using KDD dataset is higher as compared to Quadratic- SVM. But, the Quadratic- SVM using NSL-KDD dataset is higher as compared to SONN algorithm. In Specificity matrix for Quadratic- SVM using KDD dataset is higher as compared to SONN algorithm. Further, the Quadratic-SVM using NSL-KDD dataset is higher as compared to SONN algorithm. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But, Quadratic-SVM using NSL-KDD dataset is higher compared to SONN. For G-mean matrix, SONN algorithm using KDD dataset is higher compared to Quadratic-SVM. But in the NSL-KDD dataset, Quadratic- SVM is higher than SONN.

In Set-2, for Precision matrix using KDD dataset, Quadratic- SVM is higher than SONN algorithm. But for NSL-KDD data set, Quadratic- SVM is higher as compared to SONN algorithm. In Recall matrix, SONN algorithm using KDD dataset is higher as compared to Quadratic- SVM. But using NSL-KDD data set, Quadratic-SVM is higher as compared to SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher as compared to Quadratic- SVM. But, Quadratic-SVM using NSL-KDD dataset is higher as compared to SONN algorithm. For MCC, SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But for NSL-KDD data set Quadratic- SVM is higher than SONN algorithm.

Likewise, Table- 5.3 found that, in Set-1, the Overall Accuracy matrix for SONN algorithm using KDD dataset is higher compared to Cubic-SVM. But, the Cubic-SVM using NSL-KDD dataset is higher compared to SONN algorithm. In Specificity matrix, for Cubic-SVM using KDD dataset is higher as compared to SONN algorithm. Again, the Cubic-SVM using NSL-KDD dataset is higher as compared to SONN. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher as compared to Cubic-SVM. But, Cubic-SVM using NSL-KDD dataset is higher as compared to SONN. For G-mean matrix, SONN algorithm using KDD dataset is higher compared to Cubic-SVM. But in NSL-KDD dataset, Cubic-SVM figures higher as compared to SONN.

In Set-2, for Precision matrix, using KDD dataset, Cubic-SVM is higher than SONN algorithm. But for NSL-KDD dataset, Cubic-SVM is higher than SONN algorithm. In Recall matrix, SONN algorithm using KDD dataset is higher as compared to Cubic-SVM. But, using NSL-KDD dataset, Cubic-SVM is higher as compared to SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher as compared to Cubic-SVM. But, Cubic-SVM using NSL-KDD dataset is higher as compared to SONN algorithm. For MCC, SONN algorithm using KDD dataset is higher than Cubic-SVM. But for the NSL-KDD dataset, Cubic-SVM is higher than SONN algorithm.

Again, Table-5.4 on analysis resulted that, in Set-1, the Overall Accuracy matrix for Quadratic-SVM using KDD dataset is higher as compared to SONN algorithm. But, again the Quadratic-SVM using NSL-KDD dataset is higher as compared to SONN algorithm. In Specificity matrix for Quadratic- SVM using KDD dataset is higher as compared to SONN algorithm. Further, the Quadratic- SVM using NSL-KDD dataset is higher as compared to SONN algorithm. In Sensitivity matrix, the SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But, Quadratic-SVM using NSL-KDD dataset is higher compared to SONN. For G-mean matrix, SONN algorithm using KDD dataset is higher compared to Quadratic-SVM. But in the NSL-KDD dataset, Quadratic-SVM is higher than SONN algorithm.

In Set-2, for Precision matrix using KDD dataset, SONN algorithm is higher than Quadratic-SVM. But for the NSL-KDD dataset, Quadratic-SVM is higher as compared to SONN algorithm. In Recall matrix, SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But using NSL-KDD dataset, Quadratic- SVM is higher as compared to SONN algorithm. For F-Measure, again SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But Quadratic-SVM using NSL-KDD dataset is higher as compared to SONN algorithm. For MCC, SONN algorithm using KDD dataset is higher as compared to Quadratic-SVM. But for the NSL-KDD dataset, SONN algorithm is higher than Quadratic-SVM.

After the experiments, it is observed that the Support Vector Machine provided the best result for all the performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and MCC as compared to Artificial Neural Network for NSL-KDD dataset.

To enhance the performance of the Support Vector Machine with Artificial Neural Network, we adopted a hybrid method where the robust Lagrangian support vector machine (RLSVM) with an online learning algorithm was used with a modified kernel matrix. In RLSVM, consider a binary classification problem, the training set is $T = \{(\mathrm{xi}, \mathrm{yi})\}_{i=1}^{l}$, where $x_i \epsilon\ \mathbb{R}^n$ is the instance feature, and $y_i = \pm 1$ is the instance label of x_i. The optimization problem of standard SVM in the linear case is:

$$\begin{aligned} &\min_{w} = \frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{l} \xi_i \\ \text{S.t.}\quad & y_i\big((w.x_i) + b\big) + \xi_i \geq 1 \qquad (1)\\ & \xi_i \geq 0, i = 1, \dots, l \end{aligned}$$

Where C > 0 is the penalty parameter.

Based on the standard SVM, Lagrangian support vector machine (LSVM) Mangasarian and Musicant (2001) made two minor changes:

$$\begin{aligned} &\min_{w,b,\xi} = \frac{1}{2} (\parallel w \parallel^2 + b^2) + \frac{C}{2} \xi^T \xi \\ \text{S.t.}\quad & y_i\big((w.x_i) + b\big) + \xi_i \geq 1 \qquad (2)\\ & i = 1, \dots, l \end{aligned}$$

The first change is that LSVM seeks for the decision hyperplane in an n+1-dimension space instead of n dimension space. This improvement has been investigated in (Mangasarian and Musicant, 1999 & Lee and Mangasarian, 2001). Second, LSVM use 2-norm for the regulation of error ξ to avoid the nonnegative constraint of ξ_i. By bringing in lagrangian multiplier vectors $\alpha_i \geq 0$. The dual problem is:

The dual problem is an unconstrained optimization problem.

5.10 Online Algorithm for Lagrangian Support Vector Machine

Let A = $[x_1, x_2, \dots, x_l]^T$, D = diag($y_1, y_2, \dots, y_l$), and then we can get that Q = $\frac{1}{C}$ + HH^T, where H = D[A - e], e = $(1, \dots, l)^T$.

The Karush-Kuhn-Tucker(KKT) condition for the optimization problem is: α . (Qα - e) = 0 and α ≥ 0, Qα – e ≥ 0. By the identity that 0 ≤ a ⊥ b ≥ 0 ⇔ a = (a - γb)$_+$, $\gamma \geq 0$, the iteration scheme for LSVM is

$$\alpha^{i+1} = Q^{-1}(e + ((Q\alpha^{i} - e) - \gamma\alpha^{i})_{+}) i = 0, 1, ... , \gamma > 0 \quad (3)$$

Where (x), denotes the negative components of vector x are set to zero. While 0 $<\gamma<\frac{2}{C}$, the algorithm can achieve global linear convergence from any start point (Mangasarian and Musicant, 2001). Based on LSVM, online LSVM (OLSVM) is proposed (Duan *et al.*, 2009). OLSVM adds the samples which don't satisfy KKT condition ($y_i f(x_i) < 1$) into a training set to learn. Assuming that there are already k samples in the current training set, and the newly added sample is x_{k+1}. The corresponding dual problem after adding x_{m+1} is:

$$\begin{matrix} min \\ (\alpha, \alpha_{k+1}) \end{matrix} \; \frac{1}{2}(\alpha^{T}\alpha_{k+1})Q_{new}\binom{\alpha}{\alpha_{k+1}} - e^{T}\binom{\alpha}{\alpha_{k+1}} \quad (\alpha^{T}\alpha_{k+1}) \geq 0 \quad (4)$$

Let h = $y_{k+1}(X_{K+1}^{T}$, -1), and then $H_{new} = \binom{H}{h}$, $Q_{new} = \frac{1}{C} + \binom{H}{h}\ (H^{T}h^{T})$. Similar to the formulation (3), the solution of a new optimization problem is

$$\alpha_{new}^{i+1} = Q_{new}^{-1}(e + ((Q_{new}\alpha_{new}^{i} - e) - \sigma\alpha_{new}^{i})_{+})\ i = 0, 1, . . . , \sigma > 0 \quad (5)$$

Where $\alpha_{new}^{0} = \binom{\alpha}{0}$, and α is the optimal solution to the problem before increment new samples. According to Eq. 5, it can be observed that the key problem of the iteration scheme is to solve Q_{new}^{-1}. Based on some lemmas and theorem (Yuan, and Sun, 1997 & Biggio *et al.*, 2011). Q_{new}^{-1} can be calculated by:

$$Q_{new}^{-1} = C\,(I - \binom{H}{h}(B - \frac{Bh^{T}hB}{1+hBh^{T}})(H^{T}h^{T})) \quad (6)$$

Where B = $(\frac{1}{C} + H^{T}H)^{-1}$, is the consequence of the late step of the calculation.

Algorithm 1: The online learning algorithm for LSVM.

Step 1. Initialization. Give parameter C, γ such that 0 <γ<$\frac{2}{C}$ and the precision ϵ> 0, obtain the classifier C_1 by training T and let k =1.

Step 2. Get new data (x_{N+k} , y_{N+k}), let α_{m+k} = 0, i =0

Step 3. Check whether new data satisfies the KKT conditions

3.1. If KKT conditions are satisfied, (x_{N+k}, y_{N+k}) is deleted and C_k is invariant, k = k+1, and repeat step 2 till all the instances have been trained.

3.2. If KKT conditions are satisfied, add the new data into the training set, and then compute the inversion of the new matrix Q according to Eq. (6).

Step 4. Compute α_{new}^{i+1} according to Eq. (3).

Step 5. If$\| \alpha_{new}^{i+1} - \alpha_{new}^{i} \| \leq \epsilon$, get new classifier $C_{k+1,}$ k = k+1 and repeat step 2 till all the instances have been trained. Otherwise, i = i+1 and repeat step 4.

5.11 Robust Online learning algorithm with Lagrangian Support Vector Machine (ROLALSVM)

This section presents the method to improve the robustness of SVMs, a simple kernel matrix modification is used. The modification has been studied using for SVM in batch learning (Biggio *et al.*, 2011). In this chapter, its extension in online learning is explored. Firstly, a set of Boolean random variables $\varepsilon_i \in$ [0, 1] (i = 1, . . . , l) are introduced to indicate whether the corresponding labels y_i are reversed. Then replace y_i with y_i' = y_i(1 – 2ε_i) which means that if ε_i = 1, y_i' = $-y_i$, the label is flipped, otherwise ε_i = 0, y_i' = y_i i.e. not label flipped. In the dual problem of LSVM (3), the matrix Q = $y_i y_j K(x_i x_j)$ which is related to y_i will be influenced by the wrong label. Considering the label noise, the matrix can be rewritten as Q = $y_i y_j K(x_i x_j)(1 - 2\varepsilon_i)(1 - \varepsilon_j)$. Assuming that every label has the same probability to be flipped, then the variables ε_i, i = 1, . . . , l are independent identical distributed and it's mean σ^2 = $\mu(1 - \mu)$. Under the assumption, the expected value of Q can be computed by

$$E(Q_{ij}) = \begin{cases} y_i y_j K(x_i . x_j) & if\ i = j \\ y_i y_j K(x_i . x_j)(1 - 4\sigma^2) & otherwise \end{cases} \qquad (7)$$

The expected value E (Q_{ij}) is still a positive semi-definite kernel matrix. The matrix Q is replaced with E(Q), which can be regarded as a matrix correction to improve the robustness of the online LSVM model against the label noise. This method is a heuristic formula, that is, it may not get the optimal solution of the model with labelled noise. But it can modify the classifier which is badly affected by the noise, and get a better classifier. The dual problem of LSVM (3) with kernel matrix modification which is called robust Lagrangian Support Vector Machine (RLSVM) can be written as

$$\min_{0 \le \alpha \epsilon R^l} \frac{1}{2} \alpha^T (Q \circ R)\alpha - e^T \alpha \qquad (8)$$

Where the elements R_{ij} of matrix R are given by

$$R_{ij} = \begin{cases} 1 & if\ i = j \\ 1 - 4\sigma^2 & Otherwise \end{cases} \qquad (9)$$

If G = $Q \circ R$, the RLSVM dual problem can be simplified as

$$\min_{0 \le \alpha \in R^L} \frac{1}{2} \alpha^T G\alpha - e^T \alpha \qquad (10)$$

It could clearly found that (10) has the same form with (3), only replacing the kernel matrix Q with the modified matrix G. G can be computed by G = $\frac{1}{C}$ + (1 - 4σ^2) HH^T, according to Q = $\frac{1}{C}$ + HH^T. If P = $\sqrt{1 - 4\sigma^2}H$, then P^T = $\sqrt{1 - 4\sigma^2}H^T$ and G = $\frac{1}{C}$ + PP^T. Similarity, using the optimization KKT condition of the problem (10), RLSVM algorithm obtains the iterative formula:

$$\alpha^{i+1} = G^{-1}(e + ((G\alpha^i - e) - \sigma\alpha^i)_+) \quad i = 0, 1, \ldots, \sigma > 0 \qquad (11)$$

Robust Online learning algorithm with Lagrangian Support Vector Machine (ROLALSVM)

Assuming that the new income data stream isT_j= $\{(x_i, y_i)\}_{i=1}^{l_j}$(j= 1, 2, . . .). After the inclusion of new instances, the corresponding dual problem (10) should be

$$\min_{\alpha_{new}\geq 0} \frac{1}{2}\alpha_{new}^T G_{new}\alpha_{new} - e^T\alpha_{new} \tag{12}$$

Where $\alpha_{new} = \begin{pmatrix}\alpha \\ \bar{\alpha}\end{pmatrix} = \begin{pmatrix}\alpha \\ 0_{l_j\times 1}\end{pmatrix}$, α is the optimal solution before the increment and the initial solution $\overline{\alpha}$ corresponding to a new sample set T_j takes 0 of each component. Similar with (3), the iteration scheme for the solution of a new problem (12) is:

$$\alpha_{new}^{i+1} = G_{new}^{-1}(e + ((G_{new}\alpha_{new}^i - e) - \sigma\alpha_{new}^i)_+) \quad \text{i = 0, 1, . . . , } \sigma > 0 \tag{13}$$

From the above iteration formula, it can find that the key is to compute G_{new}^{-1}. The incremental computation of G_{new}^{-1} has two cases: linear and nonlinear. Here, the inverse of matrix G is computed using the SMW identity which converses the computation of m (the number of instances) order inverse matrix to the computation of n+1 (the dimension of instance) order inverse matrix. This method makes the online algorithm suitable for large-scale datasets and reduces computation consumption. The SMW identity is:

$$(\frac{1}{\tau} + AA^T)^{-1} = \tau(I - A(\frac{I}{\tau} + A^T A)^{-1}A^T \tag{14}$$

Where $\tau > 0$, A is an m $\times n$ matrix. Let matrix $\bar{A}$ denotes the feature matrix of new instances, and $\overline{D}$ denotes the label matrix. Then according to the online algorithm of LSVM, $\bar{P} = \sqrt{1-4\sigma^2}\bar{H} = \sqrt{1-4\sigma^2}\bar{D}$ [$\bar{A}$- e], $P_{new} = \begin{pmatrix}P \\ \bar{P}\end{pmatrix} = \begin{pmatrix}\sqrt{1-4\sigma^2}\bar{D}\ [\bar{A} - \text{ e}] \\ \sqrt{1-4\sigma^2}\bar{D}\ [A - \text{ e}],\end{pmatrix}$, and the corrected kernel matrix $G_{new} = \frac{1}{C}$ + (1-$4\sigma^2)H_{new}H_{new}^T$. According SMW identity,

$$G_{new}^{-1} = \frac{1}{C} + P_{new}P_{new}^T = \text{C(I - }P_{new}(\frac{I}{C} + P_{new}^T P_{new})^{-1}P_{new}^T) \tag{15}$$

$$= \text{C (I - }P_{new}(\frac{I}{C} + P^T P + \bar{P}^T\bar{P})^{-1}P_{new}^T)$$

The mini-batch online learning algorithm for <u>ROLALSVM</u> is shown in Algorithm 2 (below).

1. To initialise. C is the parameter, γ such that 0 <γ<$\frac{2}{C}$, $\epsilon > 0$ and initial training set $T_0 = \{(x_i, y_i)\}_{i=1}^m$.

Obtain the classifier C_0 (let k =0) by training T_0.

2. Get new dataset T_j = $\{(x_i, y_i)\}_{i=1}^{l_j}$ (j = 1, 2, . . .).
3. Check whether new data satisfies the KKT conditions. For every data $(x_i, y_i) \in T_j$ (1, 2, . . . , l_j)

3.1 If KKT conditions are satisfied, $(x_i, y_i) \in T_j$is deleted and $l_j = l_j - 1$, else KKT conditions are not satisfied, keep the new data in a dataset T_j.

3.2 Add the new data set T_j to the training set and let α_{m+l_j} = 0, I = 0. Then, compute the inversion of the new matrix G.

4. Compute $\alpha_{new}^{i=1}$.
5. If$\| \alpha_{new}^{i+1} - \alpha_{new}^{i} \| \leq \epsilon$get the new classifier C_k, k = k+1 and repeat step 2 till all the instances have been trained. Otherwise, I = i+1 and repeat step 4.

5.12 Performance Evaluation

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition:

$$Acccuracy = \frac{Number\ of\ correct\ predictions}{\text{Total number of predictions}}$$

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

Where *TP* = True Positives, *TN* = True Negatives, *FP* = False Positives, and *FN* = False Negatives.

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

Training dataset:

Table 5.5 Confusion Matrix Table for training dataset

	Predicted Yes	**Predicted No**
Actual Yes	47618	1
Actual No	0	42372

Accuracy = 99.99%

Testing dataset:

Table 5.6 Confusion Matrix Table for testing dataset

	Predicted Yes	**Predicted No**
Actual Yes	5287	4
Actual No	377	4331

Accuracy= 96.19 %

Thus, the analysis of Confusion Matrix table for training dataset and Confusion Matrix table for testing dataset placed in Table- 5.5 and 5.6 found the accuracy 99.99% and 96.19% respectively.

5.13 Conclusion

The performance comparison of the Self-Optimizing Neural Network (SONN) algorithm and both Quadratic and Cubic-SVM classification techniques using both KDD and NSL-KDD (training and test) datasets have experimented with Set-1 and Set-2 in both Table-5.2 and Table- 5.3. Here, to know the performance of different classifiers over the dataset, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G- Mean, Precision, Recall, F-Measure, and MCC were used. The experiments resulted that, Support Vector Machine performed the best result for all the performance matrices compared to Artificial Neural Network for the NSL-KDD dataset.

Again, to enhance the performance of the Support Vector Machine with Artificial Neural Network, we adopted a hybrid method where the robust Lagrangian support vector machine (RLSVM) with an online learning algorithm was used with a modified kernel matrix. The performance evaluation ‘Accuracy’ which is a metric for evaluating classification models on training dataset and testing dataset found to be 99.99 % and 96.19 % respectively.

CHAPTER-6

SUMMARY AND CONCLUSION

Chapter-6

Summary and Conclusion

6.1 Summary

Intrusion Detection System (IDS) involves a clustering mechanism not only to identify the illegitimate entry of intruders to the system which may cause-effect to the hardware, software, or both that controls the system in a network environment but also to detect the perpetrators before causing impairment to the data, resources, etc. (Vinchurkar and Reshamwala, 2012). For smooth, faultless, consistency of system operations especially in a network domain, IDS has become an indispensable component as it not only defends to impairment of the system but also restrained from undesirable aggress, irrational, inconsistent behavior of the computer. Added to these dimensions, it also allows a conducive environment by protecting the system from various operations in a network domain, audit network, and system configurations.

It could be found from the overall discussions from **Chapter-1** that, security has become indispensable in the network environment as every operation in all sectors is being performed on the Internet. It is more prominent to save the data and other information from the attackers. Therefore, the intrusion detection system has become crucial as it is a defense mechanism that identifies the malicious action targeted at computing and networking resources.

Artificial Neural Network and Support Vector Machine are both two recognized data mining techniques that are mostly used to detect intrusion. Universally, an Artificial Neural Network comprising of various techniques has been accepted as a viable mechanism for anomaly detection. It comprises various techniques to detect intrusion. Broadly, it is grouped into three categories such as, (i) Supervised ANN-based intrusion detection, (ii) Unsupervised ANN-based intrusion detection, and (iii) Hybrid ANN-based intrusion detection with various application platforms. While, the Supervised Artificial Neural Network to Intrusion Detection System relates to Multi-Layer Feed-Forward (MLFF) Neural Networks, Multilayer Perceptron (MLP), and Recurrent Neural Networks, the unsupervised Artificial Neural Network concerns with Self-Organising Map (SOM) and the hybrid application is a combination of both the supervised and unsupervised Artificial

Neural Network to detect intrusion (Sodiya *et al.*, 2014). However, it restricts two aspects, (i) Lower detection precision, especially for low-frequent attacks i.e. Remote to Local (R2L), User to Root (U2R), and (ii) Weaker Detection Stability (Kashyap *et al.*, 2013).

SVM, on the other hand, developed in the reverse order to the development of neural networks (NNs) to deliver a unique solution as the optimality problem is convex. It is evolved from the sound theory to the implementation and experiments against NNs which follow the heuristic path from applications and extensive experimentation to the theory Wang (2005). SVM is a linear machine that was developed by Vapnik (1995) and it solves the problems related to classification, learning, and prediction (Kausar *et al.*, 2011). As compared to other classifiers, SVM adopts the principle of structural risk minimization (SRM), it avoids the local minimum and solves issues like learning, and provides good generalization ability. It performs the classification of the data vectors by a hyperplane or set of hyperplanes in a high-dimensional space. For classification, there can be several hyperplanes for separation but the best hyperplane produces maximum margin between the data points of two classes. In many cases, the data points are not linearly separable in the input space. So, they need nonlinear transformation into a high dimensional space and then the linear maximum margin classifier can be applied. Kernel functions are used for this purpose. They are used at the training time of the classifiers to select the support vectors along the surface of the function.

It was discovered from the conversations from **Chapter-2** that, Intrusion Detection System implements the detection in a network-based platform in three ways such as (i) Host-based Intrusion Detection System (HIDS), (ii)Network-based Intrusion Detection System (NIDS), and (iii) Distributed Intrusion Detection System (DIDS). The HIDS and NIDS categories are based on deployment while, DIDS offers both Host and Network-based kind of protection (Snapp *et al.*, 1991). The Intrusion Detection System which is grouped into several types based on the type of systems it monitors focuses on observing uncongenial, incorrect, and anomalous activities in the network-based system. Further, it not only implies a process of monitoring the events that occur in a computer system or network environment but also examines the perceptible indication of potential threats computer security violation. It is a software application that is employed for network and/or information monitoring systems to detect vicious activities.

The whole neural network experiments were performed by using Knowledge Extraction based on Evolutionary Learning (KEEL), which is an open-source (GPLv3) Java software available at (http://sci2s.ugr.es/keel/dataset.php) and it is used for a large number of different knowledge data discovery tasks (http://www.keel.es/). Windows 7 Home Basic (64-bit) was used as the testbed operating system, Intel(R) Core (TM) i3 CPU M 370 @ 2.40GHz processor, 3GB of RAM.

The performance of various neural network classification algorithms such as LVQ, RBFN, DECR-RBFN, EVRBFN, MLP-BP, and SONN have been evaluated based on the KDD dataset with full 41-dimension features was used throughout the experiments. To measure the performance of the different algorithms, various performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-measure, and Matthews's Correlation Coefficient (MCC) were used. Among various tested classification algorithms, the SONN algorithm shows the best result (Overall Accuracy) compared to other tested algorithms. According to the time constraints again SONN algorithm revealed the best result to train the dataset.

After the experiments, the following inferences were deduced. The experimental results of Artificial Neural Network Algorithms using KDD (Training) dataset placed in the Table- 2.9 after analysis found that the SONN algorithm has performed the best result for Overall Accuracy i.e, 98.51%. But, the algorithm EVRBFN classifies the instance with 91.30%, whereas other algorithms such as LVQ, RBFN, DECR-RBFN, and MLP-BP resulted in 89.35%, 87.49%, 84.29%, and 68.63% respectively.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e, 94.18%. But other algorithms like RBFN, LVQ, EVRBFN, and DECR-RBFN classify the dataset with 89.99%, 87.30%, 88.04%, and 81.36% respectively. But, MLP-BP algorithm classifies the training instance with 36.08%, which shows very low performance as compared to other algorithms.

The result for Sensitivity or True Positive Rate, again SONN algorithm shows the best performance i.e. 99.22%. The algorithms like MLP-BP and EVRBFN show 98.12% and 94.26% respectively and it is also considered as better performance for Sensitivity as compared with SONN. But the other algorithms like LVQ, DECR-RBFN, and RBFN and performed 91.20%, 86.94%, and 85.22% respectively.

For the result of G-mean or G-measure, SONN algorithm performed the best i.e. 96.67% among others. So, here both Specificity (94.18%) and Sensitivity (99.22%) of SONN are high then the G-mean of SONN (96.67%) is also relatively high. Further, the G-mean value for EVRBFN constitutes 91.10% because the value of both Specificity (88.04%) and Sensitivity (94.26%) are also superior. But the performances of the other algorithms like LVQ (89.23%), RBFN (87.57%) and DECR-RBFN (84.11%) not very good. Finally, the performance of MLP-BP (59.50%) is very low as compared to other algorithms; because its Specificity value is much lower i.e. 36.08%.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 99.05% of exactness or quality of the classifier. But the other algorithms like EVRBFN (96.25%) of exactness, whereas RBFN (90.38%), LVQ (88.80%), DECR-RBFN (83.74%), and MLP-BP (62.88%) of exactness. Ultimately it deduced that MLP-BP algorithm shows very low performance as compared with others.

The result of Recall or Sensitivity or True Positive Rate (TPR), again SONN algorithm shows the best performance i.e. 99.22%. The algorithms like MLP-BP and EVRBFN show 98.12% and 94.26% respectively and it is also considered as better performance for Sensitivity as compared with SONN. But the other algorithms like LVQ, DECR-RBFN, and RBFN performed 91.20%, 86.94%, and 85.22%

respectively which otherwise means that RBFN algorithm shows very low performance.

The result of F-measure shows that SONN algorithm performed the best result i.e. 99.13% because the result for both Precision and Recall is relatively very high. i.e. 99.05% and 99.22% respectively. The result for EVRBFN algorithm shows 97.09%. The result of LVQ algorithm comes with 89.98% because of the result of Precision i.e. 88.80%. Subsequently, for RBFN, DECR-RBFN, and MLP-BP algorithms, the result of F-measure comes with 87.72%, 85.31%, and 76.64% respectively.

The result of MCC for MLP-BP algorithm shows the best result among all other algorithms i.e. 94.61% while, SONN shows 93.54% for MCC. But the other algorithms like LVQ, EVRBFN, RBFN, DECR-RBFN algorithms show very low performance i.e. 21.27%, 21.17%, 20.49%, and 18.90% respectively.

The experimental results of Artificial Neural Network using KDD (Test) dataset placed in Table-2.10 found that SONN algorithm has performed the best result for Overall Accuracy i.e. 96.34%. But the algorithm, RBFN classifies the instance with 87.23%, whereas other algorithms such as DECR-RBFN, LVQ, EVRBFN, and MLP-BP algorithms resulted in 83.92%, 82.16%, 73.36%, and 68.30% respectively.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e. 95.39%. But other algorithms like RBFN classify the dataset with 89.47% whereas, DECR-RBFN, LVQ, and EVRBFN classify the dataset with 81.84%, 75.01%, and 63.50% respectively which shows very low performance. But, MLP-BP algorithm classifies the test instance with 35.51%, which also reveals very low performance as compared to other algorithms.

The result for Sensitivity or True Positive Rate, again MLP-BP algorithm shows the best performance i.e. 98.01%. The algorithms like SONN and LVQ show 97.40% and 88.63% respectively and it is also considered as better performance for

Sensitivity as compared with MLP-BP. But the other algorithms like DECR-RBFN, RBFN, and EVRBFN performed 85.80%, 85.22%, and 82.29% results respectively. Finally, EVRBFN algorithm shows very low performance.

For the result of G-mean or G-measure, SONN algorithm performed the best result i.e. 96.39% among others. Further, the G-mean value for RBFN constitutes 87.32% because the value of both Specificity (89.47%) and Sensitivity (85.22%) is also higher. But the performances of the other algorithms like DECR-RBFN, (83.79%), LVQ (81.54%), EVRBFN (72.29%) which is not very good. Finally, the performance of MLP-BP (59.00%) is very low as compared to other algorithms.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 95.03% of exactness or quality of the classifier. But the other algorithms like RBFN (90.03%) of exactness, whereas DECR-RBFN (83.91%), LVQ (79.65%), EVRBFN (71.33%), and MLP-BP (62.65%) of exactness. Ultimately it deduced that MLP-BP algorithm shows very low performance compared to other algorithms.

The result for Recall denotes the percentage of the retrieved objects means, it shows the completeness or quantity of the algorithm. So, here the result of Recall or Sensitivity or True Positive Rate (TPR) in MLP-BP algorithm shows the best performance i.e. 98.01%. The other algorithms like SONN and LVQ show 97.40% and 88.63% respectively and it is also considered as better performance for Sensitivity as compared with MLP-BP. But the other algorithms like DECR-RBFN, RBFN, and EVRBFN performed 85.80%, 85.22%, and 82.29% results respectively, which otherwise means that EVRBFN algorithm shows very low performance.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here SONN algorithm gives the best result i.e. 96.20%. Then the result for RBFN algorithm shows 87.55% because the result for both Precision and Recall also shows very good. i.e. 90.03% and 85.22%. Then the result of DECR-RBFN and LVQ algorithm comes to 84.84% and 83.90%. But, for MLP-BP and EVRBFN algorithms, the result of F-measure comes to 76.43% and 76.41% respectively only because of their corresponding Precision value, which is very low i.e. 62.65% and 71.33%.

The result of MCC for SONN algorithm shows the best result among all other algorithms i.e. 30.64% while, RBFN algorithm shows 25.18% for MCC. But the other algorithms like DECR-RBFN, LVQ, EVRBFN, MLP-BP algorithms show very low performance i.e. 23.16%, 21.93%, 17.23%, and 11.48% respectively.

The experimental results of Artificial Neural Network Algorithms using NSL- KDD (Training) dataset in Table 3.5 shows that SONN algorithm has performed the best result for Overall Accuracy i.e., 96.28%. But the algorithm, RBFN classifies the instance with 91.11%, whereas other algorithms such as DECR-RBFN and MLP-BP algorithms resulted in 84.15% and 59.55% respectively. It means MLP-BP algorithm performs very low as compared with others. Again, LVQ and EVRBFN show very low performance i.e. 59.95% each.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e. 96.52%. But other algorithms like RBFN classify the dataset with 91.67% whereas, DECR-RBFN with 81.53%. But, MLP-BP algorithm classifies the training instance by 15.76%, which also reveals very low performance as compared to other algorithms. But, both LVQ and EVRBFN algorithm performed 0.0%.

The result for Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm shows the best performance i.e. 100% each. The other algorithms like SONN show 96.01% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like MLP-BP, DECR-RBFN, RBFN performed 87.55%, 54.42%, and 53.93% results respectively.

For the result of G-mean or G-measure, SONN algorithm performed the best result i.e. 96.26% among others. Further, the G-mean value for RBFN constitutes 70.32% because the value of both Specificity (91.67%) and Sensitivity (53.93%) is also higher. But the performance of an algorithm like DECR-RBFN is 66.61%. The performance of the MLP-BP algorithm is 37.14%, which is not very good because of its Specificity value i.e. 15.76%. Finally, the performance of LVQ and EVRBFN comes with 0.0% each.

The result of Precision or Positive Predictive Value, again SONN algorithm reveals the best result among all other algorithms i.e. 96.09% of exactness or quality of the classifier. But the other algorithms like RBFN (90.38%) of exactness, whereas DECR-RBFN (83.74%), MLP-BP (62.88%), LVQ (52.46%), EVRBFN (52.40%) of exactness. Ultimately it deduced that MLP-BP algorithm shows very low performance as compared with other algorithms.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, the resulting Recall or Sensitivity or True Positive Rate, LVQ, and EVRBFN algorithm show the best performance i.e. (100%) each. The other algorithms like SONN show 96.01% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like MLP-BP, DECR-RBFN, and RBFN performed 87.55%, 54.42%, and 53.93% results respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here SONN algorithm gives the best result i.e. 96.05%. Then, the result for MLP-BP algorithm shows 73.19% because the result for both Precision and Recall also shows very good. i.e. 62.88% and 87.55%. Then, the result of LVQ and EVRBFN algorithm comes to 68.81% and 68.76% respectively. But, for RBFN and DECR-RBFN algorithms, the result of F-measure comes to 67.55% and 66.04% respectively only because of their corresponding Recall value, which is very low i.e. 53.93% and 54.52%.

The result of MCC for MLP-BP algorithm shows the best result among all other algorithms i.e. 94.61% while, the SONN algorithm also shows a very good result for MCC i.e. (82.58%). But the other algorithms like RBFN, DECR-RBFN, LVQ, EVRBFN algorithms show very low performance i.e. 20.49%, 18.91%, 17.17%, and 17.17% respectively.

Chapter-3 of the study incorporates the detailed experiments using neural network algorithms. The whole neural network experiments are done by using Knowledge Extraction based on Evolutionary Learning (KEEL) open-source software. Among various neural network algorithms and based on their performance evaluation to six algorithms such as i) LVQ, ii) RBFN, iii) DECR-RBFN, iv) EVRBFN, v) MLP-BP, and vi) SONN were chosen for experiments. To measure the performance of these algorithms, different performance matrices are also used. The various matrices include, i) Overall accuracy, ii) Specificity, iii) Sensitivity, iv) G-Mean (Kubat *et al.,* 1998) v) Precision, vi) Recall, vii) F-Measure (Lewis and Gale, 1994) and viii) Matthews's Correlation Coefficient (MCC). To test different classification algorithms of Artificial Neural Network, NSL-KDD dataset which is an enhanced edition of the original KDD'99 dataset is used for the whole experiments consisting of 41 features including 25190 training instances.

The experimental results Artificial Neural Network using NSL- KDD (test) dataset in Table 3.6 shows that SONN algorithm has performed the best result for Overall Accuracy i.e. 95.91%. But the algorithm, RBFN classifies the instance with 91.14%, whereas other algorithms such as DECR-RBFN and MLP-BP algorithms resulted in 83.64% and 59.23% respectively. Then the other algorithm like LVQ and EVRBFN the result of Overall Accuracy comes to 53.17% and 52.95% correspondingly. It means EVRBFN algorithm performs very low as compared with others.

The result for Specificity or True Negative Rate, again SONN algorithm performed the best result i.e. 96.52%. But other algorithms like RBFN classify the dataset with 88.69% whereas, DECR-RBFN with 81.16%. But, MLP-BP algorithm classifies the test instance with 01.98%, which also reveals very low performance as compared to other algorithms, whereas, both LVQ and EVRBFN algorithm comes with 0.0%.

The result for Sensitivity or True Positive Rate, LVQ and EVRBFN algorithm shows the best performance i.e. 100% each, whereas, the other algorithm like MLP-BP shows 87.84% and it is also considered as better performance for

Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like SONN, DECR-RBFN, RBFN performed 67.29%, 54.34%, and 54.21% results respectively.

For the result of G-mean or G-measure, RBFN algorithm performed the best result i.e. 69.34% reasonably high among others, because it is a geometric mean of both Specificity and Sensitivity. Further, the G-mean value for DECR-RBFN constitutes 66.41% because the value of both Specificity (81.16%) and Sensitivity (54.34%) is also higher. But the performance of an algorithm like SONN is 46.92%, which is not very good because of its Sensitivity value i.e. 67.29%, even if, its Specificity value is so high i.e.96.52%. The performance of the MLP-BP algorithm is 13.18%, which is very low performance. Finally, the performance of LVQ and EVRBFN comes to 00% each because the Specificity value of each algorithm is also 0.0%.

The result of Precision or Positive Predictive Value, DECR-RBFN algorithm reveals the best result among all other algorithms i.e. 97.94% of exactness or quality of the classifier. But the other algorithms like SONN (97.59%) of exactness, whereas RBFN (90.03%), MLP-BP (62.65%), EVRBFN (52.56%), LVQ (52.46%) of exactness. Ultimately it deduced that LVQ algorithm shows very low performance as compared with other algorithms.

The result for Recall denotes the percentage of the retrieved objects means it shows the Completeness or quantity of the algorithm. So, the result of Recall or Sensitivity or True Positive Rate, LVQ and EVRBFN algorithm shows the best performance i.e. 100% each, whereas, the other algorithm like MLP-BP shows 87.84% and it is also considered as better performance for Sensitivity as compared with LVQ and EVRBFN. But the other algorithms like SONN, DECR-RBFN, and RBFN performed 67.29%, 54.34%, and 54.21% results respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here SONN algorithm gives the best result i.e. 79.65%. Then the result for MLP-BP algorithm shows 73.13% and the result of DECR-RBFN and LVQ algorithm comes to 69.89% and 68.97% respectively. But, for EVRBFN and RBFN algorithms, the result of F-measure comes to 68.90% and 67.67% respectively.

The result of MCC for SONN algorithm shows, the best result among all other algorithms which comes to 30.65%. Surprisingly, the RBFN algorithm also shows very good performance for MCC i.e. 25.18%. But the other algorithms like DECR-RBFN, LVQ, EVRBFN, and MLP-BP show very low performance i.e. 21.12%, 19.17%, 19.07%, and 11.48% respectively.

Chapter-4 of the study explains the detailed experiments using various linear and non-linear SVM classifiers. Among various linear and non-linear SVM classifiers and based on their performance evaluation top six SVM classifiers such as Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, Coarse Gaussian SVM algorithms were chosen for experiments. To measure the performance of these algorithms, different performance matrices are also used. The various matrices include, i) Overall Accuracy, ii) Specificity, iii) Sensitivity, iv) G-Mean (Kubat *et al.*, 1998) v) Precision, vi) Recall, vii) F-Measure (Lewis and Gale, 1994) viii) Matthews's Correlation Coefficient (MCC), and ix) ROC Curve (AUC). The whole Support Vector Machine experiments are performed using Matrix Laboratory (MATLAB) Version 8.50.197613 (R2015a) with 64-bit (win 64) and Lic no. 161052 (Bryant and Garber, 1999 & Sumathi and Paneerselvam, 2010) open-source software. Windows 7 Home Basic (64-bit) as the testbed operating system, Intel(R) Core (TM) i3 CPU M 370 @ 2.40GHz processor, 3GB of RAM were used. To test different classification algorithms of SVM, both original KDD'99 and NSL-KDD dataset were used.

The experimental results of Support Vector Machine using KDD (Training) dataset placed in Table- 4.4 found that Quadratic SVM and Cubic SVM have performed the best result for Overall Accuracy i.e. 98.50% each as compared to other classifiers. But the classifiers like Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 98.30% each followed by both Quadratic SVM and Cubic SVM, which shows very good performance. But the other classifiers like Linear

SVM and Coarse Gaussian SVM also show good performance for the result of Overall Accuracy i.e. 97.70% and 97.50% respectively.

The result for Specificity or True Negative Rate, both Quadratic SVM and Cubic SVM performed the best result i.e. 99.89% each. But other classifiers like Fine Gaussian SVM and Medium Gaussian SVM classify the dataset with 99.83% and 99.19% correspondingly. But, Coarse Gaussian SVM classifies the training instance with 97.43% and Linear SVM classifies the training instance with 97.00%, which also shows very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Medium Gaussian SVM performed the best result for Sensitivity (99.49%), whereas, Fine Gaussian SVM and Linear SVM shows 98.73% and 98.51% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian SVM performed 97.60%. It also shows a very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e. 88.08% and 87.92% respectively.

For the result of G-mean or G-measure, the Medium Gaussian SVM performed the best result i.e. 99.34%, because it is a geometric mean of both Specificity and Sensitivity. Further, the G-mean value for Fine Gaussian SVM constitutes 99.28%, which is slightly less performance than Medium Gaussian SVM. But the performances of the other classifiers like Linear SVM (97.75%) and Coarse Gaussian SVM (97.51%) also show good performance. Then, finally, Cubic SVM and Quadratic SVM also illustrate good performance as compared to others. i.e. 93.79% and 93.71% correspondingly.

The result of Precision or Positive Predictive Value, Medium Gaussian SVM performed the best result i.e. 99.54% of the exactness or quality of the classifier. But the other classifiers like Cubic SVM and Quadratic SVM, both show 99.16% of exactness. Then Fine Gaussian SVM and Linear SVM show 98.88% and 98.65% of exactness. Finally, Coarse Gaussian SVM (97.86%), which shows slightly fewer performance than other classifiers.

The result of Recall or Sensitivity or True Positive Rate (TPR), again Medium Gaussian SVM performed the best result i.e. 99.49%, whereas, the other classifiers like Fine Gaussian SVM (98.53%) and Linear SVM (98.51%), which is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. But the other classifiers like Coarse Gaussian SVM (97.60%). It also shows a very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e. 88.08% and 87.92% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here again, Medium Gaussian SVM performed the best result i.e. 99.51%. Then the result for Fine Gaussian SVM (98.80%) shows very well. Then the result of Linear SVM comes with 98.58% which also shows very good performance, because of the result of Precision i.e. 98.65%, and Recall i.e. 98.51% which is a comparatively very good performance. But the classifier like Coarse Gaussian SVM shows 97.72%. It also shows a very good performance. Finally, Cubic SVM and Quadratic SVM also show good performance as compared to others. i.e. 93.29% and 93.20% respectively.

The result of MCC for both Quadratic SVM and Cubic SVM shows the best result among all other algorithms i.e. 92.72% and 92.64% respectively, whereas, the other classifiers shows very low performance i.e. Fine Gaussian SVM (24.55%), Medium Gaussian SVM (24.39%), Linear SVM (24.38%) and finally Coarse Gaussian SVM shows 23.69% for MCC, which is considered as very low performance among other classifiers. The Fine Gaussian SVM is shown preeminent results for ROC Curve among all the classifiers i.e. 99.98%.

The experimental results of Support Vector Machine using KDD (Test) dataset placed in Table-4.5 shows that Quadratic SVM has performed the best result for Overall Accuracy i.e. 96.91% as compared to other classifiers. But the Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 96.22% and 95.31% respectively followed by Quadratic SVM, which shows very good performance. But the other classifiers like Linear SVM, Coarse Gaussian SVM, and Cubic SVM also show good performance for the result of Overall Accuracy i.e. 93.14%, 90.59%, and 83.11% respectively.

The result for Specificity or True Negative Rate, Coarse Gaussian SVM has performed the best result i.e. 99.74%. But other classifiers like Fine Gaussian SVM classify the dataset with 99.26%, which also shows very good performance. But Quadratic SVM classifies the instance with 98.93% followed by Medium Gaussian SVM, which shows the result with 98.70%. Then other classifiers like Linear SVM and Cubic SVM reveal the result with 98.39% and 90.64% respectively, which is also considered as very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Fine Gaussian SVM performed the best result for Sensitivity is 88.01%, whereas, Quadratic SVM and Medium Gaussian SVM shows 87.83% and 80.02% respectively and it is also considered as better performance for Sensitivity as compared with Fine Gaussian SVM. Then Coarse Gaussian SVM performed 97.60%. It also shows a very good performance. Finally, Linear SVM, Coarse Gaussian SVM, and Cubic SVM show very low performance as compared to others. i.e. 69.47%, 49.35% and 49.21% respectively.

For the result of G-mean or G-measure, the Fine Gaussian SVM performed the best result i.e. 93.47%. Further, the G-mean value for Quadratic SVM constitutes 93.21%, because the value of both Specificity (98.93%) and Sensitivity (87.83%) are also superior, which is slightly less performance than Fine Gaussian SVM. But the performances of the other classifiers like Medium Gaussian SVM (88.87%) and Linear SVM (82.68%) also show good performance. Then finally Coarse Gaussian SVM and Cubic SVM reveal not very good performance as compared to others. i.e. 70.16% and 66.79% correspondingly.

The result of Precision or Positive Predictive Value, Coarse Gaussian SVM performed the best result i.e. 97.70% of exactness or quality of the classifier followed by Fine Gaussian Support Vector Machine which is constituted with 96.34% of exactness. But the other classifiers like Quadratic SVM and Medium

Gaussian SVM both show 94.78% and 93.18% of exactness respectively. Then Linear SVM and Cubic SVM show 90.55% and 53.84% of exactness. So, finally, Cubic SVM shows very low performance for Precision than other classifiers.

The result of Recall or Sensitivity or True Positive Rate (TPR), Fine Gaussian SVM performed the best result for Sensitivity is 88.01%, whereas, Quadratic SVM and Medium Gaussian SVM shows 87.83% and 80.02% respectively and it is also considered as better performance for Sensitivity as compared with Fine Gaussian SVM. Then Coarse Gaussian SVM performed 97.60%. It also shows a very good performance. Finally, Linear SVM, Coarse Gaussian SVM, and Cubic SVM show very low performance as compared to others. i.e. 69.47%, 49.35% and 49.21% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Fine Gaussian SVM performed the best result i.e. 91.99%. Then the result for Quadratic SVM shows very good performance followed by Fine Gaussian SVM i.e.91.17%. The result of Medium Gaussian SVM comes with 86.10%, which is also considered a very good performance. But the classifier like Linear SVM shows 78.62% due to its corresponding Recall value i.e. 69.47%, which shows low performance. Finally, Coarse Gaussian SVM and Cubic SVM both are show slightly low performance as compared to others. i.e. 65.58% and 51.42% respectively, because the Recall value of both the classifiers is 49.35% and 49.21% respectively.

The result of MCC for Cubic SVM shows the best result among all other algorithms i.e. 93.08%. Whereas, the other classifiers show very low performance i.e. Fine Gaussian SVM (18.23%), Quadratic SVM (18.13%), Medium Gaussian SVM (16.48%), Linear SVM (14.26%), and finally Coarse Gaussian SVM shows 10.27% for MCC, which is considered as very low performance among other classifiers. The Quadratic SVM is shown preeminent results for ROC curves among all the classifiers i.e. 99.04%.

The experimental results of Support Vector Machine using NSL-KDD (training) dataset placed in Table-4.8 shows that Quadratic SVM and Cubic SVM have performed the best result for Overall Accuracy i.e. 100.00% each as compared to other classifiers. But the Fine Gaussian SVM and Medium Gaussian SVM classify the instance with 99.40% each, which shows very good performance. But the other classifiers like Linear SVM and Coarse Gaussian SVM also show good performance for the result of Overall Accuracy i.e. 97.90% and 97.50% respectively.

The result for Specificity or True Negative Rate, both Quadratic SVM and Medium Gaussian SVM performed the best result i.e. 99.74% each. But other classifiers like Cubic SVM and Linear SVM classify the dataset with 99.60% and 99.06% correspondingly, which is also considered as very good performance. But, Fine Gaussian SVM and Coarse Gaussian SVM classify the training instance with 98.89% and 97.64% respectively, which also shows very good performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Fine Gaussian SVM performed the best result for Sensitivity (99.81%), whereas, Cubic SVM, Quadratic SVM, and Medium Gaussian SVM show 99.53%, 99.45%, and 99.15% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian SVM performed 97.37%. It also shows a very good performance. Finally, Linear SVM also shows good performance as compared to others. i.e. 96.82%.

For the result of G-mean or G-measure, the Quadratic SVM performed the best result i.e. 99.59%, because it is a geometric mean of both Specificity and Sensitivity. Further, the G-mean value for Cubic SVM constitutes 99.56%, which is slightly less performance than Quadratic SVM. But the performances of the other classifiers like Medium Gaussian SVM (99.44%) and Fine Gaussian SVM (99.34%) also show very good performance. Then finally Linear SVM and Coarse Gaussian SVM also illustrate good performance as compared to others i.e. 97.93% and 97.50% correspondingly.

The result of Precision or Positive Predictive Value, Quadratic SVM performed the best result i.e. 100% of exactness or quality of the classifier. But the classifiers like Medium Gaussian SVM and Cubic SVM also show the best result i.e. 99.77% and 99.64% of exactness or quality respectively as compared to Quadratic SVM. Then Linear SVM and Fine Gaussian SVM show 99.15% and 99.03% of exactness, which is considered as the better result for Precision. Finally, Coarse Gaussian SVM (97.89%), which shows slightly fewer performance than other classifiers.

The result of Recall or Sensitivity or True Positive Rate (TPR), Fine Gaussian SVM performed the best result for Sensitivity (99.81%), whereas, Cubic SVM, Quadratic SVM, and Medium Gaussian SVM show 99.53%, 99.45%, and 99.15% respectively and it is also considered as better performance for Sensitivity as compared with Medium Gaussian SVM. Then Coarse Gaussian SVM performed 97.37%. It also shows a very good performance. Finally, Linear SVM also shows good performance as compared to others. i.e. 96.82%.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Quadratic SVM performed the best result i.e. 99.72%. The result for Cubic SVM (99.58%) shows very good performance because the result for both Precision and Recall also shows very good. Then the result of Medium Gaussian SVM comes with 99.45%, which is also considered a very good performance. But the classifier like Fine Gaussian SVM shows 99.41% followed by Medium Gaussian SVM. It also reveals a very good result for F-measure. Finally, Linear SVM and Coarse Gaussian SVM also show good performance as compared to others. i.e. 97.97% and 97.62% respectively.

The result of MCC for Quadratic SVM shows the best result i.e. 99.59% followed by Cubic SVM, which also shows very good result i.e. 99.12% while, the other classifiers shows very low performance i.e. Medium Gaussian SVM (24.64%), Fine Gaussian SVM (24.59%) followed by both Linear SVM and Coarse Gaussian SVM i.e. 23.89% and 23.68% respectively, which is considered as very low performance as compared to both Quadratic SVM and Cubic Support Vector

Machine. The Fine Gaussian SVM found to be excellent results for ROC Curve among all the classifiers i.e. 99.98%.

The experimental results of Support Vector Machine using NSL-KDD (test) dataset placed in Table- 4.9 found that Quadratic SVM has performed the best result for Overall Accuracy i.e. 99.60% as compared to other classifiers. But the Medium Gaussian SVM and Fine Gaussian SVM classify the instance with 99.25% and 98.84% respectively, which shows very good performance. But the other classifiers like Linear Support Vector Machine, Coarse Gaussian SVM, and Cubic SVM also show good performance for the result of Overall Accuracy i.e. 97.40%, 96.89%, and 44.85% respectively.

The result for Specificity or True Negative Rate, Quadratic SVM has performed the best result i.e. 99.74% each. Medium Gaussian SVM classifies the negative classes with 99.87%, which is also considered as very good performance. The classifiers like Linear SVM and Fine Gaussian SVM classify the dataset with 99.68% and 99.17% correspondingly, which is also considered as very good performance. But, Coarse Gaussian SVM and Cubic SVM classify the test instance with 98.24% and 03.58% respectively. It means that Cubic SVM shows very low performance as compared to other classifiers.

The result for Sensitivity or True Positive Rate, Cubic SVM performed the best result for Sensitivity (99.39%), whereas, Quadratic SVM shows 99.16% result for Sensitivity, which is considered as very good performance as compared with Cubic SVM. Then Medium Gaussian SVM and Fine Gaussian SVM performed 98.42% and 98.41% respectively which also shows very good performance. Finally, Coarse Gaussian SVM and Linear SVM also show good performance as compared to others i.e. 95.11% and 94.39% respectively.

For the result of G-mean or G-measure, the Quadratic SVM performed the best result i.e. 99.55%, because it is a geometric mean of both Specificity and Sensitivity. Further, the G-mean value for Medium Gaussian SVM constitutes 99.14%, which is slightly less performance than Quadratic SVM. But the performances of the other classifiers like Fine Gaussian SVM (98.79%) and Linear SVM (97.00%) also show very good performance. Then finally Coarse Gaussian SVM illustrates good performance as compared to others i.e. 96.67%. But the Cubic SVM shows very low performance than others i.e. 18.86%.

The result of Precision or Positive Predictive Value, Quadratic SVM performed the best result i.e. 99.91% of the exactness or quality of the classifier. But the classifiers like Medium Gaussian SVM and Linear SVM also show a very good performance i.e. 99.83% and 99.56% of exactness or quality respectively as compared to Quadratic SVM. Then Fine Gaussian SVM and Coarse Gaussian SVM show 98.92% and 97.62% of exactness, which is also considered as the better result for Precision. Finally, Cubic SVM comes with 43.82%, which shows very low performance than other classifiers.

The result for Recall or Sensitivity or True Positive Rate, Cubic SVM performed the best result for Sensitivity (99.39%) whereas, Quadratic SVM shows 99.16% result for Sensitivity, which is considered as very good performance as compared with Cubic SVM. Then Medium Gaussian SVM and Fine Gaussian SVM performed 98.42% and 98.41% respectively which also shows very good performance. Finally, Coarse Gaussian SVM and Linear SVM also show good performance as compared to others. i.e. 95.11% and 94.39% respectively.

The result of F-measure always depends on the result of both Precision and Recall, because it is the harmonic mean of both Precision and Recall. So, here Quadratic SVM performed the best result i.e. 99.54%. Then the result for Medium Gaussian SVM is 99.12%, which shows very good performance. Then the result of Fine Gaussian SVM comes with 98.66%, which is also considered a very good performance. Then the classifier like Coarse Gaussian SVM and Linear SVM comes with 96.35% and 96.91% results respectively. It also reveals a very good result for F-measure. Finally, Cubic SVM shows low performance as compared to others.

The result of MCC for Cubic-SVM shows the best result i.e. 44.29%. But the other classifiers show very low performance as compared to Cubic-SVM, i.e.

Quadratic-SVM (12.34%), Medium Gaussian SVM (12.24%), and Fine Gaussian SVM (12.19%). Then finally both Linear SVM and Coarse Gaussian SVM show very low-performance i.e. 11.72% and 11.64% respectively. The Quadratic SVM was found to be preeminent results for ROC Curve among all the classifiers i.e. 99.91%.

Chapter-5 describes the detailed experiments about the comprehensive comparison among the best performer classifiers. The experimental results deduced the following inferences. Table-5.2 deduced that SONN algorithm performed the best result using the KDD training dataset for Overall Accuracy (98.51%), Sensitivity (99.22%), and G-mean (96.67%) respectively. For Specificity, Quadratic-SVM is found to be the best i.e. 99.89%. For Precision, Quadratic-SVM performed better than SONN i.e. 99.16%. SONN performed the best result using the KDD dataset for Recall (99.22%), F-Measure (99.13%), and MCC (93.54%) respectively.

It was found that Quadratic-SVM performed the best for Overall Accuracy i.e. 100% while using the NSL-KDD training dataset as compared to SONN algorithm. Again, Quadratic-SVM performed the best result for other performance matrices, such as Specificity (99.74%), Sensitivity (99.45%), and G-mean (99.59%).

Again, Quadratic-SVM performed the best result in using NSL-KDD dataset for the performance matrices i.e. Precision (100%), Recall (99.45%), F-Measure (99.72%), and MCC (99.59%).

It was observed from Tabel-5.3 that, SONN algorithm performed the best result using the KDD training dataset for Overall Accuracy (98.51%), Sensitivity (99.22%), and G-Mean (96.67%) as compared to Cubic- SVM. Further, Cubic-SVM performed the best result for Specificity (99.89%) and Precision (99.16%). Finally, it could be deduced that SONN shows the best performance for Recall (99.22%), F-Measure (99.13%), and MCC (93.54%).

On using of NSL-KDD training dataset and compared to SONN algorithm, Cubic-SVM shows the best result for all the performance matrices i.e. Overall

Accuracy (100%), Specificity (99.60%), Sensitivity (99.53%), G-Mean (99.56%), Precision (99.64%), Recall (99.53%), F-Measure (99.58%) and MCC (99.12%).

Table-5.4 deduced that for both KDD and NSL-KDD test dataset again Quadratic-SVM is found to be the best among all other Support Vector Machine classifiers and SONN algorithm is found to be the best among all other Artificial Neural Network algorithms. After using of KDD test dataset and compared it to SONN algorithm, Quadratic-SVM performed the best result for both Overall Accuracy (96.91%) and Specificity (98.93%). But for Sensitivity and G-Mean, SONN algorithm performs the best result i.e. 97.40% and 96.39% respectively.

Again, for Precision, Recall, F-Measure, and MCC, SONN algorithm performed the best result i.e. 95.03%, 97.40%, 96.20%, and 30.64% respectively. After using of NSL-KDD test dataset and compared to SONN algorithm, Quadratic-SVM performed the best result for all the performance matrices i.e. Overall Accuracy (99.60%), Specificity (99.93%), Sensitivity (99.16%), G-Mean (99.55%), Precision (99.91%), Recall (99.16%) and F-Measure (99.54%). But for MCC, SONN algorithm showed the best result i.e. (30.65%).

6.2 Conclusion

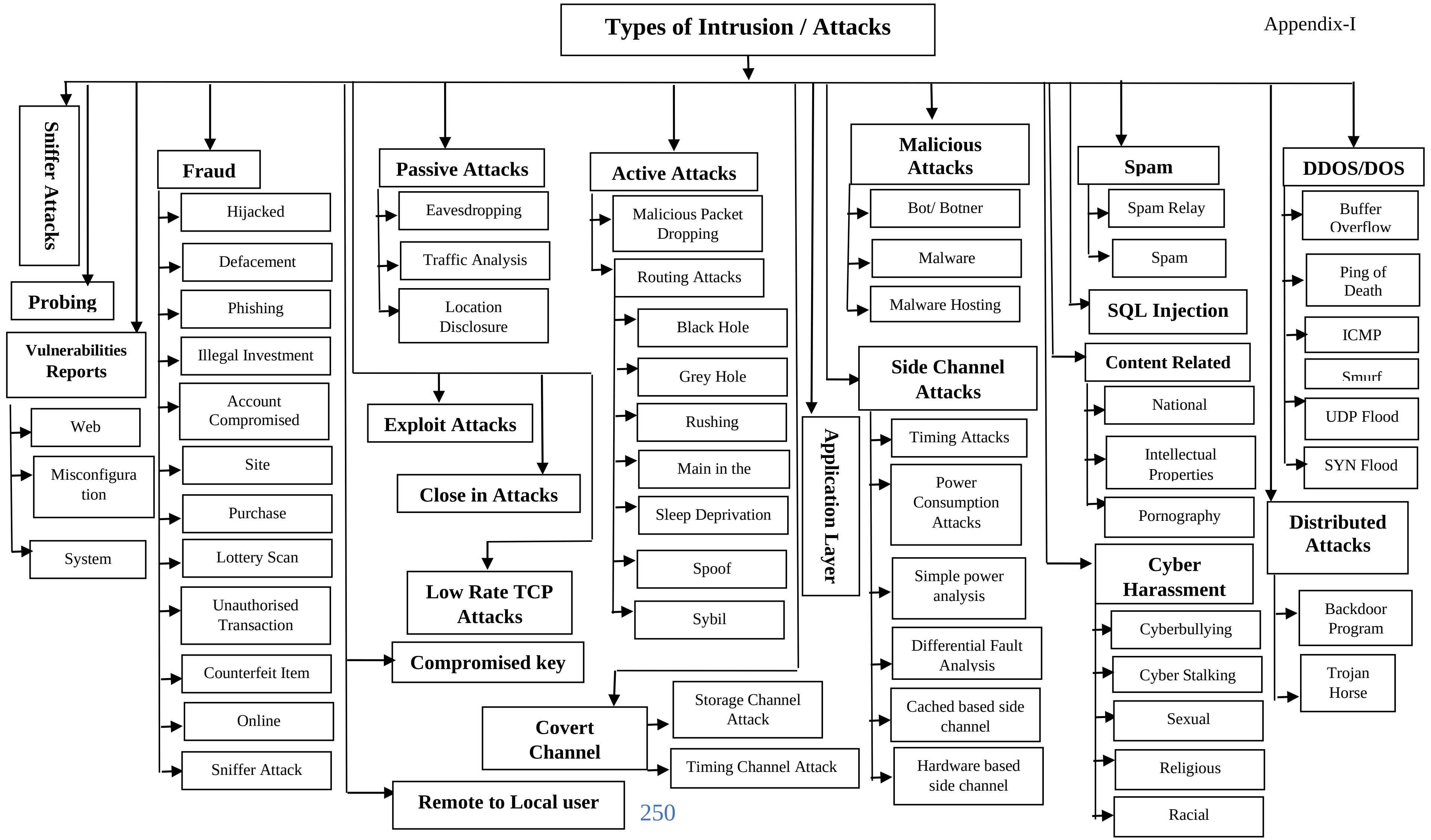
After all the experiments ultimately, it is observed that Support Vector Machine performed the best result for all the performance matrices such as Overall Accuracy, Specificity, Sensitivity, G-Mean, Precision, Recall, F-Measure, and MCC as compared to Artificial Neural Network for NSL-KDD dataset.

Again, to enhance the performance of the Support Vector Machine with Artificial Neural Network, we adopted a hybrid method where the Robust Lagrangian Support Vector Machine (RLSVM) with an online learning algorithm is used with a modified kernel matrix. The performance evaluation ‘Overall Accuracy’ which is a metric for evaluating classification models on training dataset and testing dataset found to be 99.99% and 96.19% respectively.

6.3 Future Scope of the Study

The present study and the evaluated results open avenues for further research on various techniques from both the Artificial Neural Network family and Support Vector Machine family in a more scientific method kernel selection, data pre-processing, feature selection, etc. The future study can also be concentrated on multi-class classification, application of Support Vector Machine for regression issues. The future experiments can also be performed using feature reduction techniques with other classification techniques such as the C4.5 algorithm, ID3 algorithm, K-nearest neighbour classifiers algorithm, Naive Bayes Algorithm, etc. in Intrusion Detection System.

Appendix-I



Source: Anwar *et al.* (2017). From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions, *Algorithms*. 10, 39: 1-24. doi: 10.3390/ a 10020039

Comparative Statement of Types of Attacks with Examples

Attacks	**Examples**	**Description/ Objectives**	**Parameters**		
			Confidentiality	**Integrity**	**Availability**
Insider	U2R, Flooding Attacks, Port scanning	Damaging the network or system by an authorized user	Yes	No	Yes
Flooding	DOS, DDOS, Direct and Indirect DOS	Blockage of the machine through invalid information	No	No	Yes
DOS	Buffer overflow, ICMP, Ping of death, Smurf, UDP fold, SYN flood	Creating a barrier to the availability of resources to the target users by the attackers.	No	No	Yes
Port Scanning	TCP scanning, UDP scanning, SYN scanning, ACK scanning, Windows scanning	Finding open port closed port and filtered port from the list of open ports by the attacker to block the services.	No	No	Yes
Application Layer Attacks or Host Based Attacks	Spamming, Race condition attacks, Buffer overflow attacks, Man-in-the-middle attacks	Causing faults by the attacker in the application layer or operating system of the server.	No	Yes	Yes
Passive Attacks	Eavesdropping, Traffic analysis, Location disclosure	Creating a disturbance by the attacker in the performance and network operation including locating information.	Yes	Yes	No

Active Attacks	Routing attacks, malicious traffic dropping	Causing interruption by the attacker through bringing on malicious code, alternating information, and damage to the network.	Yes	Yes	Yes
Routing Attacks	Spoofing attacks (IP and URL spoofing) Rushing, Gray hole, Blackhole, Cybil, Sleep Deprivation	The attackers modify the routing protocol in the mobile ad-hoc network.	Yes	Yes	Yes
Code Red Attacks		The exploitation of an acknowledged vulnerability in Microsoft IIS servers.	No	Yes	Yes
Side-Channel Attacks		Extracting personal information from the systems.	Yes	Yes	Yes
Covert Channel Attacks		Using a covert channel, the attacker extracts confidential information from another's system.	Yes	Yes	Yes
Adversarial Attacks against IDS		Disable functioning of IDRS and affect the detection accuracy of IDS.	No	No	Yes

Source: Anwar *et al*. (2017). From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions, *Algorithms*. 10, 39: 1-24. DOI: 10.3390/ a 10020039

APPENDICES

Appendix-III

List of Dreadful Latest Virus

Sl. No	**Date of release**	**Name of virus**	**Technical Name**	**Type**	**Intensity**	**Description/ Effect**
1	7.9.2015	Android/Leech.A		Malware	N/A	A malware program available on Google Play that looks like a legitimate game (BrainTest) for the user but, after installation, decrypts a malicious payload. it persistently runs with full privileges to execute any other malicious code that could include thief identity, unwanted payments via SMS, spying functionality, and others because the code is dynamically loaded from a command and control server.
2	28.8.2015	Generic.e! 71CDC3201116		Virus	Low	Searches local drives, removable and network shares for Windows PE executable files to infect. Replaces the original entry point of the files it infects with its viral code and appends itself to the last section of the PE image
3	28.8.2015	Generic.e		Trojan	Low	Searches local drives, removable and network shares for Windows PE executable files to infect. It replaces the original entry point of the files it infects

						with its viral code and appends itself to the last section of the PE image.
4	27.8.2015	Exploit/Stagefright.M		PUP	N/A	Detects media files that attempt to exploit a vulnerability (CVE-2015-1539) in the Android media library Stagefright.
5	9.3.2013	Android/MetOrg SMS.A		PUP	Low	It is a PUP that sends SMS messages during playing the game.
6	16.3.2011	Regin!sys		Trojan	Low	The most common installation methods involve system or security exploitation, and unsuspecting users manually executing unknown programs. Distribution channels include e-mail, malicious or hacked Web pages, Internet Relay Chat (IRC), peer-to-peer networks, etc.
7	15.5.2006	Hoots. A	W32/Hoots.A	Worm	Medium	Attempts to send an image of a snowy owl to the local network printers. It spreads through shared resources and mapped drives.
8	11.5.2006	KittyKat.A	W32/KittyKat.A	Trojan	Medium	No destructive effects but aims to spread inserting its code in files with a RAR extension.

9	10.5.2006	MS06-020	MS06-020	Vulnerability	Low	Vulnerabilities in Macromedia Flash Player,
10	10.5.2006	MS06-019	MS06-019	Vulnerability	Low	A critical vulnerability in *Exchange Server* 2003/ 2000, which allows hackers to gain remote control of the affected computer with the same privileges as the logged-on user.
11	10.5.2006	MS06-018	MS06-018	Vulnerability	Low	If exploited successfully, *MS06-018* allows hackers to cause the computer to stop responding.
12	07.05.2006	Nabload. CW	Trj/Nabload.CW	Trojan	Medium	Reaches the computer in an executable file that passes itself off as a *Windows Media Player* file.
13	04.05.2006	Downloader.ITW	Trj/Downloader. ITW	Trojan	Low	*Downloader.ITW* does not spread automatically using its means but needs an attacking user's intervention to reach the affected computer.
14	01.05.2006	Banker.CTD	Trj/Banker.CTD	Trojan	Medium	Monitors if the user accesses websites belonging to certain banking entities, to obtain passwords. Then, the gathered information is sent to an email address.

15	01.05.2006	Hiviti. A	Bck/Hiviti.A	Backdoor	Medium	Logs the keystrokes typed by the user, which allows it to obtain sensitive information, such as passwords. Then, it sends the gathered data to a certain email address.
16	01.05.2006	Nugache. A	W32/Nugache	Worm	Medium	Opens several ports and connects to certain IP addresses, to receive remote P2P commands. It exploits the vulnerabilities LSASS and RPC DCOM, to spread to as many computers as possible, among other means of transmission.
17	26.04.2006	Briz. F	Trj/Briz.F	Trojan	Low	Obtains confidential data from the affected computer and prevents users from accessing websites belonging to certain antivirus companies.
18	26.04.2006	Banker.CSC	Trj/Banker.CSC	Trojan	Low	Monitors if the user accesses websites belonging to certain banking entities, to obtain passwords. Then, the gathered information is sent to its author.
19	24.04.2006	Crazyfrog.A	W32/CrazyFrog.	Worm	Medium	Obtains the *MSN Messenger* passwords and the banking data of the affected users if they access websites belonging to banking entities. It spreads via *MSN Messenger*.

20	24.04.2006	Lootseek.AU	Trj/Lootseek.AU	Trojan	Low	Downloads the *Trj/Rizalof.BL* to the affected computer, and it ends several processes belonging to security tools and Windows updates.
21	20.4.2006	Bifrose. KV	Bck/Bifrose.KV	Backdoor	Medium	Allows gaining remote access to the affected computer and captures certain information entered or saved by the user, with the corresponding threat to privacy.
22	19.04.2006	Goldun. IL	Trj/Goldun.IL	Trojan	Medium	Obtains the access data to the *e-gold* account of the affected user. It has been massively sent via email as an attached file.
23	18.04.2006	HarBag.A	Trj/HarBag.A	Trojan	Medium	Harvests email addresses from the affected computer and then sends them to a certain URL. It does not spread automatically by its means.
24	17.04.2006	Lootseek. AG	Bck/Lootseek.AG	Backdoor	Medium	Connects to an IRC server to receive remote control commands, such as download and run files, and it ends several processes belonging to security tools and Windows updates.
25	11.4.2003	Nimrod. B	W32/Nimrod.B	Worm	Medium	Spreads and affects other computers and does not spread automatically using its' means.

Source: http://home.mcafee.com/virusinfo, http://www.pandasecurity.com/india/homeusers/security-info.

Bibliography

Abraham, A., Grosan, C. and Vide, C.M. (2007). Evolutionary design of intrusion detection programs, *International Journal of Network Security*. 4(3): 328-339.

Acid, S. and de Campos L.M. (2003). Searching for Bayesian Network Structures in the space of Restricted Acyclic Partially Directed Graphs, *Journal of Artificial Intelligence Research*. 8:445-490.

Aggarwal, P. and Sharma, S. K. (2015). Analysis of KDD Dataset Attributes-Class wise for Intrusion Detection, *Procedia Computer Science*. 57. 842-851.

Ahdi, F. ,Khandani, M.K. , Hamedi, M. and Haghani, A. (2012). Traffic Data Collection and Anonymous Vehicle Detection Using Wireless Sensor Networks. University of Maryland.1-52.

Ahmed, J., Jafri, M. N., Ahmad, J. and Khan, M.I.(2007). Design and Implementation of a Neural Network for Real-Time Object Tracking. World Academy of Science, Engineering and Technology, *International Journal of Computer, Information, Systems and Control Engineering*. 1(6):1825-1828.

Ahmed, M., Mahmood, A. and Rafiqul, I. Md. (2016). A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*. 55: 278-288.

Akande, K.O. , Owolabi, T.O. , Twaha, S. and Olatunji, S.O.(2014).Performance Comparison of SVM and ANN in Predicting Compressive Strength of Concrete, *IOSR Journal of Computer Engineering*. 16(5): 88-94. F

Alcala-Fdez, J., Fernandez, A., Luengo, J., Derrac, J., Garcia, S., Sanchez, L., and Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework, *Journal of Multi-Valued Logic & Soft Computing*.17: 255–287.

Alex, M. and Andrew (2000). An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods by Nello Christianini and John Shawe-Taylor. Cambridge University Press, Cambridge. xiii+ 189: 687-689.

Allen, J., Christie, A., Fithen, W., McHugh, J., Pickel, J., Stoner, E., Ellis, J., Hayes, E., Marella, J. and Willke, B. (1999). State of the Practice of Intrusion Detection Technologies, Carnegie Mellon University, Pittsburgh, PA Technical Report CMU/SEI-99-TR-028.

Al-mamory, S. O. and Jassim, F. S. (2015). On the designing of two grains levels network intrusion detection system, *Karbala International Journal of Modern Science*.1(1):15-25.

Alsadhan, A. and Khan, N. (2013). A Proposed Optimized and Efficient Intrusion Detection System for Wireless Sensor, World Academy of Science, Engineering and Technology, *International Journal of Electrical, Robotics, Electronics and Communications Engineering*.7(12): 1145-1148.

Ambwani, T. (2003). Multi class SVM implementation to intrusion detection, *Proceedings of the IEEE International Joint Conference on Neural Networks*. 3: 2300-2305.

Amorso, E. (1999). Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back ,Traps and Response. First Edition AT & T Inc.

Anderson, P. J. (1980). Computer Security Threat Monitoring and Surveillance.http://seclab . cs. ucdavis.edu/projects/history/ papers/ ande80 . pdf.

Aneetha, A.S. and Bose, S. (2012). The combined approach for anomaly detection using neural networks and clustering techniques, *Computer Science & Engineering*. 2(4): 37- 46.

Anifowose, F.A. and Eludiora, S.I. (2012).Application of Artificial Intelligence in Network Intrusion Detection, *Journal of World Applied Programming*. 2(3): 158-166.

Anwar, S., Zain, J. M., Zolkipli, M.F., Inayat, Z., Khan, S, Anthony, B., and Chang, V. (2017). From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions, *Algorithms*. 10, 39: 1-24. doi: 10.3390/ a 10020039

APCERT Annual Report. (2013). http://www.apcert.org/documents/pdf/APCERT_ Annual_ Report _2013(FINAL).pdf.

Atiya, A. and Parols, A. (2000). New Results on Recurrent Network Training. Unifying the Algorithms and Accelerating Convergence, *IEEE Trans. Neural Networks.* 11(3): 697-709.

Auria, L. and Moro, R. A., (2008). Support Vector Machines (SVM) as a Technique for Solvency Analysis. DIW Berlin Discussion Paper No. 811. http://dx.doi.org/10. 2139/ssrn.1424949.

Axelson, S. (1999). Research in Intrusion-Detection Systems: A Survey, TR: 98-17 http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.2.3547.

Axelsson, S. (2000). A preliminary attempt to apply detection and estimation theory to intrusion detection. Technical Report 00-4, Chalmers Univ. of Technology, Goteborg, Sweden.

Axelsson, S. (2000). Intrusion Detection Systems: A Survey and Taxonomy, Dept. of Computer Engineering, Chalmers University Technical Report. 99-15.

Axelsson, S. and Sands, S. (2006). Understanding Intrusion Detection through visualisation, US, Springer-Verlag.

Babatope, L.O., Babatunde, L. and Ayobami, I.(2014). Strategic Sensor Placement for Intrusion Detection in Network-Based Intrusion Detection System, *International Journal on Intelligent Systems and Applications*.02: 61-68.

Bace, R. and Mell, P.M. (2001). Intrusion Detection System, NIST Special Publication on Intrusion Detection Systems. http://www. albany.edu/acc/ courses / ia/ acc661/ sp800-31.pdf.

Bae, C., Yeh, W.C., Shukran, W.M., Chung, Y.Y. and Hsieh, T.J.(2012). A novel anomaly-network intrusion detection system using ABC algorithms, *International Journal of Innovative Computing, Information and Control*. 8(12): 8231-8248.

Bajaj, K and Arora, A. (2013). Improving the Intrusion Detection using Discriminative Machine Learning Approach and Improve the Time Complexity by Data Mining Feature Selection Methods, *International Journal of Computer Applications.* 76 (1): 5-11.

Bamakan, S. M. H., Wang, H., Yingjie, T. and Shi, Y. (2016). An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization, *Neurocomputing*.199: 90-102.

Bambrick, N. and Aylien (2016). Support Vector Machines: A simple explanation. https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation. html.

Bangal, B.C. (2009). Artificial Neural Networks. http://shodhganga.inflibnet.ac.in/ 74-100.

Barker, J., Stevens, D. and Bloom, S.F. (2001). Spread and prevention of some common viral infections in community facilities and domestic homes, *Journal of Applied Microbiology*. 91(1):7-21.

Barman, D.K. and Khataniar, G.(n.d.). Design of intrusion detection system based on Artificial neural network and application of Rough set,*International Journal of Computer Science & Communication Networks*. 2(4): 548-552.

Barto, A.G. and Sutton, R. (1997). Introduction to Reinforcement Learning.Cambridge, MIT Press.

Bauer, M.D. (2003). Building Secure Servers with Linux, Sebastopol, O'Reilly & Associates, Inc.

Bejtlich, R. (2005). The Tao of Network Security Monitoring: Beyond Intrusion Detection, Boston, Addison-Wesley.

Belavagi, M. C. and Muniyal, B. (2016). Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection, *Procedia Computer Science*. 89: 117-123.

Beniwal, S. and Arora, J. (2012). Classification and feature selection techniques in data mining, *International Journal of Engineering Research and Technology*. 1(6): 1-6.

Bezdek, J. C. and Kuncheva, L. I. (2001). Nearest prototype classifier designs: An experimental study, *International Journal of Intelligent Systems*. 16(12): 1445-1473.

Bhati, B.S. and Rai, C.S. (2016). Intrusion Detection systems and techniques: A review. *International Journal of Critical Computer-Based Systems*. 6(3): 173-190.

Bhoria, P. and Garg, K.K. (2013). Determining feature set of DOS attacks, *International Journal of Advanced Research in Computer Science and Software Engineering*. 3(5): 875-878.

Bhuyan, M., Bhattacharyya, D.K. and Kalita, J.K. (2014). Network Anomaly Detection: Methods, Systems and Tools, *IEEE Communications Surveys & Tutorials*. 16(1): 303-336.

Biba, K. J. (1977). Integrity Considerations for Secure Computer Systems, MTR-3153, The Mitre Corporation.

Biermann, E., Clote, E. and Venter, L.M. (2001). A Comparison of Intrusion Detection Systems, *Computers and Security.* 20(8): 676-683.

Biggio, B., Nelson, B., and Laskov, P. (2011). Support Vector Machines Under Adversarial Label Noise, *In.* JMLR: Workshop and Conference Proceedings 20. On Asian Conference on Machine Learning .97–112.

Bishop, M. and Venkatramanayya, S.S. (2005). Introduction to Computer Security, Delhi, Person Education. 441-442.

Bivens, A., Gupta, R., McLean, I., Szymanski, B. and White, J. (2004). Scalability and Performance of an Agent-based Network Management Middleware, *International Journal of Network Management*. 14(2): 131-146.

Blacharski, D. (1998). Network Security in a Mixed Environment, Foster city, CA, IDG Books: 275.

Boser, B.E. , Guyon, I.M. and Vapnik, V.N. (1992). A training algorithm for optimal margin classifiers, *In: Proc. 5th Annual Workshop on Computational Learning Theory, AMC press:* 144-152.

Bottou, L. and Lin, C.J. (2007). Support vector machine solvers. Large-scale kernel machines.1-27.

Boughorbel S., Jarray F., and El-Anbari M. (2017). Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLoS ONE.* 12(6): e0177678. https: // doi.org/10.1371/journal. pone.0177678.

Broomhead, D. S. and Lowe, D. (1988). *Radial basis functions, multi-variable functional interpolation and adaptive networks* (no. rsre-memo-4148). Royal signals and radar establishment malvern (United Kingdom). 1-40.

Bruha, I. (2000). From machine learning to Knowledge Discovery. Survey of preprocessing and post processing. Intelligent Data Analysis. 4: 363-374.

Bryant, M.L. and Garber, F.D. (1999). SVM classifier applied to the MSTAR public data set. AeroSense'99, *International Society for Optics and Photonics*. 3721: 1-6.

Bullinaria, J.A. (2015). Networks of Artificial Neurons, Single Layer Perceptrons Neural Computation: Lecture 3. http://www.cs.bham.ac.uk/~jxb/INC/l3.pdf.

Burges, C. J.C. (1998). A tutorial on support vector machines for pattern recognition. In. Data Mining and Knowledge Discovery. 2. Kluwer Academic Publishers, Boston: 121-167.

Caballero, A. (2009). Information Security Essentials for IT Managers Protecting Mission-Critical Systems Managing Information Security. DOI: http:/ /dx.doi. org/ 10.1016/ B978-0-12-416688-2.00001-5.

Cahyo, A.N. ,Hidayat, R. and Adhipta, D. (2016). Performance comparison of intrusion detection system based anomaly detection using artificial neural network and support vector machine,*Advances of Science and Technology for Society, Conference Proceedings of American Institute of Physics*. 1755: 070011-1–070011-7. doi: 10.1063/1.4958506.

Campos, M.M., Stengard, P.J., Milenova, B.L. (2005). Creation and Deployment of Data Mining-Based Intrusion Detection Systems in Oracle Database. 109. *In Proceedings of the 4th International Conference on Machine Learning and Applications (ICMLA 2005).*Los Angeles, USA. 97-104.

Cannady, J. (1998). Artificial Neural Network for Misuse detection. School of Computer Science and Information Sciences, Nova Southeastern University, Fort Lauderdale, FL 33314.

Carr, H. H., Snyder, C. A. and Baily, B. N. (2010). The Management of Network Security: Technology, Design and Management Control, Boston, Prentice Hall.

Caruana, R., and Niculescu-Mizil, A. (2004). Data mining in metric space: an empirical analysis of supervised learning performance criteria. *Proceedings of the10th ACM SIGKDD, International conference on Knowledge discovery and data mining* (KDD 2004), Seattle, Washington, USA.

Cecil, A. (2006). A Summary of Network Traffic Monitoring and Analysis Techniques http://www.cse.wustl.edu/~jain/cse567-06/ftp/net_monitoring.pdf.1-9.

Cerrato, I. ,Leogrande, M. and Risso, F. (2013).Filtering network traffic based on protocol encapsulation rules,*International Conference on Computing, Networking and Communications*. IEEE.1058-1063.

Chandola, V., Banerjee, A. and Kumar, V. (2007). Technical Report, TR 07-017, Minneapolis, University of Minnesota.

Chang, C.C. and Lin, C.J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* (TIST). 2(3): 27. http:// www. csie .ntu. edu.tw/~cjlin/libsvm

Chen, R., Chen, J., Chen, T., Hsieh, C., Chen, T. and Wu, K. (2005). Building an Intrusion Detection System Based on Support Vector Machine and Genetic Algorithm. In: Wang J., Liao XF., Yi Z. (*eds)* Advances in Neural Networks. 409-414. Lecture Notes in Computer Science. 3498. Springer, Berlin, Heidelberg.

Chen, W. H., Hsu, S. H. and Shen, H. P. (2005). Application of SVM and ANN for Intrusion Detection, *Computer & Operations Research.* 32: 2617-2634.

Cheng, C., Tay, W.P. and Huang, G.B. (2012). Extreme learning machines for intrusion detection, *International Joint Conference on Neural Networks (IJCNN).* 1-8: IEEE.

Cherkasova, L. ,Ozonat, K. , Mi, N., Symons, J. and Smirni, E.(2009). Automated Anomaly Detection and Performance Modelling of Enterprise Applications, *Journal of ACM Transactions on Computer Systems.* 27(3): 6.1-6.32.

Chhabra, S. India's National Cyber Security Policy (NCSP) and Organisation – A Critical assessment, *Naval War College Journal.* http://indiannavy.nic.in/ sites / default /files / nwc14/Article_6.pdf.55-70.

Cichonski , P. , Millar, T. , Grance, T. and Scarfone, K. (2012). Computer Security Incident Handling Guide. NIST Special Publication 800-61 Revision 2. U.S. Department of Commerce.

Code Red worm (2009). https://www.caida.org/research/security/ code-red/ coderedv2 _ analysis.xml.

Cohen,W. W. (1995). Fast effective rule induction, *Proceedings of 12th International conference on machine learning*, California.

Cort, A. (2004). Algorithm-based approaches to intrusion detection and response, SANS Institute Info Sec Reading Room.1-16.

Cortes, C. and Vapnik, V. (1995). Support-vector networks, *Machine learning*. 20(3): 273-297.

Cowan, C., Wagle, P., Calton Pu., Beattie, S. and Walpole, J.(2000). Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade,*Proceedings ofDARPA Information Survivability Conference and Exposition*. DISCEX '00.

Cristianini, N. and Shawe-Taylor, J. (2000). An Introduction to Support Vector Machines and other Kernel-based Learning Methods. Cambridge university press.

Curtin, M. (1997). http://www.interhack.net/pubs/network-security.pdf.1-16.

Cyber Crime Survey Report. (2014). https://www.kpmg.com/IN/en/IssuesAnd Insights/ Articles Publications/Documents/KPMG_Cyber_Crime_ survey_ report_ 2014. pdf.

Dagher, I. (2008). Quadratic kernel-free non-linear Support Vector Machine, *Journal of Global Optimisation.* 41(1):15-30.

DARPA Intrusion Detection Data sets. Cyber Systems and Technology. MIT Lincoln Laboratory, http://www.ll.mit.edu.

David, M. W. Powers. (2007). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation, Technical Report SIE-07-001, School of Informatics and Engineering. 1-24.

De La Hoz, E., Ortiz, A., Ortega, J., and Prieto, B. (2015). PCA filtering and probabilistic SOM for network intrusion detection, *Neurocomputing*, 164: 71-81.

Debar, H., Dacier, M. and Wespi, A. (1999). Towards a Taxonomy of Intrusion Detection Systems, *Computer Networks*. 31(8): 805-822

Deng, H., Zeng, Q., and Agrawal. D.P. (2003). SVM-based intrusion detection system for wireless ad-hoc networks, *In Proc. of Vehicular Technology Conference*. 2147-2151.

Denning, D. and Neumann, P. (1985). Intrusion Detection Expert System (IDES Model). *SRI* Project 6169-70, Amendment 5 to U.S. Government Contract 83F83-01-00 for SPAWAR, 15 July 1984 to 16 September 1985. http://www.csl.sri.com/ programs/ intrusion/ history.html.

Denning, D.E. (1987). An intrusion detection model, *IEEE Transactions on software engineering.* 13(2): 222-232.

Depren, O., Topallar, M., Anarim, E. And Ciliz, M.K.(2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, *Expert systems with Applications.* 29(4): 713-722.

Deshmukha, R.V. and Devadkarb, K.K. (2015). Understanding DDoS Attack & Its Effect In Cloud Environment, *Procedia Computer Science*. 49: 202-210.

Devaraju, S. and Ramakrishnan, S. (2004). Performance Comparison for Intrusion Detection System Using Neural Network with KDD Dataset,*Ictact Journal On Soft Computing.* 4(3):743-752.

Dhanabal, L. and Shantharajah, S.P. (2015). A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms, *International Journal of Advanced Research in Computer and Communication Engineering.*4(6): 446-452.

Dias, L.P. ,Crequeira, J.J.F. , Assis, K.D.R. and Almeida, R.C.(2017).Using artificial neural network in intrusion detection systems to computer networks,*Computer Science and Electronic Engineering.* IEEE. 145-150.

Dokas, P., Ertoz, L., Kumar, V., Lazarevic, A., Srivastava, J. and Tan, P.N. (2002). Data mining for network intrusion detection, *In Proc. NSF Workshop on Next Generation Data Mining*. 21-30.

Dolezelova, M. ,Muehlfeld, M. , Svistunov, M. , Wadeley, S. , Capek, T. and Hradilek, J. (2017). Red Hat Enterprise Linux 7 System Administrator's Guide, Deployment, Configuration and Administration of Red Hat Enterprise Linux 7.

Dorofki, M., Elshafie, A.H., Jaafar, O., Karim, O.A. and Mastura, S. (2012)..Comparison of Artificial Neural Network Transfer Functions Abilities to Simulate Extreme Runoff Data, *International Conference on Environment, Energy and Biotechnology, IPCBEE, IACSIT Press,* Singapore.33.39-44.

Douligeris, C. and Mitrokotsa, A. (2004). DDoS attacks and defense mechanisms: classification and state-of-the-art, *Computer Networks*. 44: 643-666. http://cys.ewi. tudelft .nl/sites/default/files/comnet.pdf.

Duan, H., Shao, X., Hou, W. He, G. and Zeng, Q. (2009). An Incremental Learning Algorithm For Lagrangian Support Vector Machines, *Pattern Recognition Letters.* 30(15) 1384–1391.

Duda, R. and Hart, P.(1973). Pattern Classification and Scene Analysis. Wiley, New York .

Duquea, S. and Omar, M.N. (2015). Using data mining algorithms for developing a model for intrusion detection system (IDS), *Procedia Computer Science*. 61: 46-51.

Easttom, C. (2011). Computer Security Fundamentals, New Delhi, Dorling Kindersley (India) Pvt. Ltd.

Eck, D. (2006).Course Lecture Notes IFT 6080, University of Montreal. http://www. iro. umontreal .ca/~pift6080/H09/documents/papers/svm_tutorial.ppt.

Einwechter,V. (2001). An Introduction to Distributed Intrusion Detection Systems. https:// www.symantec.com/connect/articles/introduction-distributed-intrusion- detection-systems.

Enterprise Applications Administration (2014). DOI: http://dx.doi.org/10.1016/ B978-0-12-407773-7.00005-3.

Farshchi, J.(2003).Intrusion Detection FAQ: Statistical based approach to Intrusion Detection. https://www.sans.org/security-resources/idfaq/statistic_ids.php.

Fawcett, T. (2006). An Introduction to ROC Analysis ,*Pattern Recognition Letters*. 27(8): 861–874. doi:10.1016/j.patrec.2005.10.010.

Folino, G. and Sabatino, P. (2016). Ensemble based collaborative and distributed intrusion detection systems: a survey, *Journal of Network and Computer Applications*. 66:1-16 http://dx.doi.org/10.1016/j.jnca.2016.03.011.

Frederick, K.K.(2001). Network Intrusion Detection Signature. http://online. Securityfocus. com/infocus/1524.

Garg, T. and Khurana, S.S. (2014). Comparison of Classification Techniques for Intrusion Detection Dataset Using WEKA, *IEEE International Conference on Recent Advances and Innovations in Engineering* (ICRAIE-2014), Jaipur, India.

Gautam, S.H. and Hari Om. (2016). Computational neural network regression model for Host based Intrusion Detection System,*Perspectives in Science*. 8: 93-95.

Gharibian, F. and Ghorbani, A.A. (2007). Comparative study of supervised machine learning techniques for intrusion detection, *InFifth* Annual Conference on Communication Networks and Services Research, *CNSR'07*.350-358. http://ieeexplore.ieee.org/ document/4215535/.

Giray, S.M. and Pola, A.G. (2013). Evaluation and Comparison of Classification Techniques for Network Intrusion Detection,*In Proc. Of the 13th IEEE International Conference On Data Mining Workshop*. Dallad: 335-342.

Goodrich, M.T. and Tamassia, R. (2011). Introduction to computer security. Boston, M.A.: Addison-Wesley.

Gorbani, A.A., Lu, W. and Tavallaee, M. (2010). Network Intrusion Detection and Prevention. Springer, New York, Dordrecht, Heidelberg, London.

Graff, M. (2001). Instrument of the information security trade. SANS Security Essentials GSEC Practical Assignment Version 1.2f.

Grim, L. (2014). IDS: File Integrity Checking. https://www.sans.org/reading-room/ whitepapers/detection/ids-file-integrity-checking-35327.

Guru Prasad. (2012). Hand Gesture Recognition Using Neural Network. http:// matlabsproj. blogspot.in/2012/06/hand-gesture-recognition-using-neural.html.

Halme, L. R. and Baue, R. K. (2012). Intrusion Detection FAQ: AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques. https://www.sans.org/security-resources/ idfaq/ aint. Php

Hand Gesture Recognition Using Neural Network (2012). http://matlabsproj.blogspot.in/ 2012/06/hand-gesture-recognition-using-neural.html.

Hans, J. and Kamber, M. (2006). Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, Burlington. 10-20. MA.

Haykin, S. (1999). Neural Networks: A comprehensive Foundation. 2nd Ed., New Delhi; Person Education (Singapore) Pvt.Ltd.

Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J. and Scholkopf, B. (1998). Support vector machines, *IEEE Intelligent Systems and their Applications*.13(4): 18-28.

Heward, G. (2014). Cryptanalysis and Attacks. http://www.experts-exchange .com/ articles/ 12460/ Cryptanalysis-and-Attacks.html.

Homma, N. and Gupta, M. M. (2002a). Superimposing Neural Learning by Dynamic and Spatial Changing Weights, *Proceedings of the 7th International Symposium, Artificial life and Robotics*.1:165-168.

Hoopes, J., Bawcom, A., Kenealy, P., Noonan,W. and Schiller, C. (2009). Virtualisation for security including Sandboxing, Disaster recovery, High availability, Forensic Analysis and Honey potting. Burlington, Elsevier, Inc.

Housley, R. (1993). Security Level Framework for Internet, https://tools.ietf.org/pdf/rfc 1457.pdf.

Howlett, R.J. and Lakhmi, C.J. (2001). Radial Basis Function Networks 2: *New Advances in Design*, Springer-Verlang, Berlin, Heidelberg.

Hoz, De la, E., Hoz,De La, E., Ortiz, A., Ortega, J. and Prieto, B. (2015). PCA filtering and probabilistic SOM for network intrusion detection, *Neurocomputing*.164: 71-81.

Hsu, C.W., Chang, C.C. and Lin, C.J. (2003). A practical guide to support vector classification. http://ntu.csie.org/~cjlin/papers/guide/guide.pdf.

http://sci2s.ugr.es/keel/datasets.php.

http://www.keel.es/

http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/index.html?/access/helpdesk/help/toolbox/nnet/neuron2.html%20

http://www.merl.com/papers/docs/TR94-03.pdf.

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/3/html/Security _Guide/s1-ids-host.html.

https://archive.ics.uci.edu/ml/datasets/KDD+ Cup+1999+Data.

https://in.mathworks.com/products/matlab. html.

https://www.statista.com/ statistics/273018/number-of-internet-users-worldwide/)

https://www.symantec.com/ content/dam/ Symantec/ docs/reports/2016-norton-cyber-security –insights comparisons -us-en. pdf.

https://www.symantec.com/ content/dam/ymantec/ docs/reports/istr-22-2017-en.pdf.

Huang, G.B., Zhou, H., Ding, X., and Zhang, R. (2012). Extreme learning machine for regression and multiclass classification, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 42(2): 513-529.

Huang, T.M. and Kecman, V. (2009). Linear Support Vector Machine. http://www.linear svm.com/

Hussain, J. and Mishra, A. (2017). An Effective Intrusion Detection Framework based on Support Vector Machine using NSL-KDD dataset,*Indian Journal of Computer Science and Engineering*. 8(6): 703-713.

Hussain, J. and Mishra, A. (2017). Performance Analysis of Some Neural Network algorithms using NSL-KDD Dataset, *International Journal of Computer Trends and Technology*. 50(1): 43-49.

Hwang, J. N., Choi, J. J., Oh, S. and Mark II, R. J. (1991). Query-based learning applied to partially trend multiplier perceptrons, *IEEE Transaction On Neural Networks.* 2: 131-136.

Hwang, T.S., Lee, T.J., and Lee, Y.J. (2007). A three-tier IDS via data mining approach, *MineNet '07 Proceedings of the 3rd annual ACM workshop on Mining network data*. San Diego, California, USA.1-6.

Ibrahim, L.M., Basheer, D.T., and Mahamod, M.S. (2013). A Comparison Study for Intrusion Database (KDD99, NSL-KDD) Based on Self Organization Map (SOM) Artificial Neural Network, *Journal of Engineering Science and Technology*. 8(1): 107-119.

Imran, H.M., Abdullah, A.B., Hussain, M., Palaniappan, S. and Ahmad, I. (2012). Intrusions Detection based on Optimum Features Subset and Efficient Dataset Selection, *International journal of Engineering and Innovative Technology (IJEIT)*. 2(6): 265-270.

Inayat, Z., Gani, A., Anuar, N.B., Khan, M.K. and Anwa, S.(2016). Intrusion response systems: foundations, design and challenges, *Journal of Network and Computer Application.* 62: 53–74. http://dx.doi.org/10.1016/j.jnca.2015.12.006.

Ingre, B. and Yadav, A. (2015). Performance Analysis of NSL-KDD dataset using Artificial Neural Network, *International Conference on Signal Processing and Communication Engineering Systems (SPACES)*. 92-96.

Internet Security Threat Report (2017). https://www.symantec.com/ content/n dam/ymantec/ docs/reports/istr-22-2017-en.pdf.

Jabez, J. and Muthukumar, B. (2015). Intrusion Detection System (IDS): Anomaly detection using outlier detection approach, *Procedia Computer Science*. 48: 338-346.

Jansen, B.J. (2006). Search log analysis: What it is, what's been done, how to do it. *Library & Information Science Research*. 28 :407–432.

Jha, J. and Ragha, L. (2013). Intrusion Detection System using Support Vector Machine,*International Journal of Applied Information Systems* (IJAIS). 25-30.

Joachims, T. (2006). Training linear SVMs in linear time, *Proc. of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.*

Jones, A. and Sielken, R. (1999). Computer System Intrusion Detection. University of Virginia Technical Report.

Jyothsna, V. and RamaPrasad, V.V. (2011). A Review of Anomaly based Intrusion Detection Systems, *International Journal of Computer Application*. 28(7): 26-35.

Kabila, R. (2008). Network Based Intrusion Detection and Prevention Systems in IP-Level Security Protocols. *World Academy of Science, Engineering and Technology International Journal of Industrial and Manufacturing Engineering*. 2(10):1152-1158.

Kabir, E. , Hu, J., Wang, H. and Zhuo, G. (2017). A novel statistical technique for intrusion detection systems, *Future Generation Computer Systems.*1-16.

Kabiri, P., Ghorbani, A.A., (2005). Research on Intrusion Detection and Response: A Survey, *International Journal of Network Security*. 1(2): 84-102.

Kahate, A. (2011). Cryptography and Network security, 3rd. Ed., New Delhi, McGraw Hill Education (India) Pvt. Ltd.

Kashyap, S., Agrawal, P., Pandey, V.C. and Keshri, S.P. (2013).Importance of Intrusion Detection System with its different approaches,*International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*. 2(5):1902-1908.

Kauser, N., Samir, B.B., Abdullah, A., Ahmad, I. and Hussain, M. (2011). A Review of classification approach using Support Vector Machine in Intrusion Detection. In: Abd Manaf A., Sahibuddin S., Ahmad R., MohdDaud S., El-Qawasmeh E. (eds) *International Conference on Informatics Engineering* *and Information Science* 2011. Part III, CCIS. 253. Springer, Berlin, Heidelberg. 24-34.

Kayacik, H.G. ,Zincir,-Heywood, N. (2005). Analysis of Three intrusion detection system Benchmark datasets using Machine learning algorithms, *Proceedings of the IEEE ISI* 2005. Atlanta, USA.

Kazienko, P. (2003). Intrusion Detection System Principles, Architecture and Measurements. HUT- Networking Laboratory.1-32. https://www.netlab.tkk.fi/opetus/s38310/02-03/ jussila_060503.pdf.

Kazienko, P. and Dorosz, P. (2003). Intrusion Detection Systems (IDS) Part I - (network intrusions; attack symptoms; IDS tasks; and IDS architecture).*WindowsSecurity.comIntrusion Detection System.*

KDD Cup 1999. Available on: http://kdd.ics.uci.edu/databases/kddcup99/ kddcup99.html.

Kecman, V. (2001). Learning and soft computing: support vector machines, neural networks, and fuzzy logic models. MIT press.

KEEL User Manual (2015). http://sci2s.ugr.es/keel/documents/KeelManual3.0.pdf.

Kemmerer, A. and Vigna, G. (2002). Intrusion detection: A brief history and overview, *Computer Security & Privacy*. 35(4): 27-30.

Kendall, K. (1999). A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems. https://www.ll.mit. edu/ideval/files/ kkendall_ thesis.pdf.

Kent, K. and Souppaya, M. (2006). Guide to Computer Security Log Management, NIST Special Publication 800-92. http://csrc.nist.gov/publications/ nistpubs/800-92 /SP800-92.pdf.

Khari, M. and Karar, A. (2013). Analysis on Intrusion Detection by machine learning techniques: A Review, *International Journal of Advanced Research in Computer Science and Software Engineering*. 3(4): 545-548.

Kim, G.H. and Spafford, E.H. (1995). The Design and Implementation of Tripwire: A File System Integrity Checker. COAST Laboratory, *Proceedings of the 2nd ACM Conference on Computer and Communications Security.*1-18.

Kissel, R. (2013). Ed., Glossary of Key Information Security Terms, National Institute of Standards and Technology, US. http://www.nist.gov/customcf/ get _pdf. cfm? pub_id = 913810.

Ko, K., Choi, S., Kang, C. and Hong, D. (2000b). RBF Multiuser Detector with Channel Estimation Capability in a Synchronosu MC-CDMA System, *IEEE Transaction on Neural Network*. 12(6):1536-1538.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection, *International Joint Conference on Artificial Intelligence*. 14(2): 1-7.

Kohavi, R. and Provost, F. (1998). On Applied Research in Machine Learning. In Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process, Columbia University, New York. Vol. 30.

Kozierok, C.M. (2003). TCP/IP Guide. http://www.tcpipguide.com/ free/t_TCP Connection Establishment Process TheThreeWayHandsh-3.html.

Krenker, A, Bester, J and Kos, A. (2011). Introduction to the Artificial Neural Networks, *In Artificial Neural Networks - Methodological Advances and Biomedical Applications.*Kenji Suzuki (Ed.) Kenji Suzuki. London: INTECH.

Kubat, M. J., Vernel, J., Rychwalski, R. W. and Kubat, J. (1998). Bulk moduli from physical aging and stress relaxation data, *Polymer Engineering & Science*. 38: 1261-1269. doi :10.1002/ pen.10296.

Kumar, S. and Yadav, A.(2014). Increasing performance of intrusion detection system, *IEEE International conference on advanced communication control and computing technology*. 546-550.

Laskov, P. , Gehl, C., Kruger, S. and Muller, K.V.(2006).Incremental support vector learning: Analysis, implementation and applications, *Journal of Machine Learning Research*. 7: 1909-1936.

Laura, A. and Moro, R. A. (2008). Support Vector Machines (SVM) as a Technique for Solvency Analysis. DIW Berlin Discussion Paper No. 811. https://ssrn.com/ abstract= 1424949 or http: // dx.doi.org/ 10.2139/ ssrn.1424949.

Lawal, O. B., Ibitola, A. and Longe, O. B. (2013). Analysis and Evaluation of Network-Based Intrusion Detection and Prevention System in an Enterprise Network Using Snort Freeware, *African Journal of Computing & ICT*. 6(2): 169-184.

Lazarevic, A., Ertoz, L., Ozgur, A., Kumar, V. and Srivastava, J. (2003). A Comparative study of Anomaly detection scheme in Network Intrusion Detection, *In: Proc. 3rd SAIM Int. Conf. On Data Mining*: 25-36.

Lazarevic, A., Kumar,V. and Srivastava, J. (2005). Intrusion Detection: A Survey. *In. Managing Cyber Threat*. Boston. Springer. 19-78.

Lee, W., Stolfo, S. J. and Mok, K.W. (1999). A data mining framework for building intrusion detection models, *Security and Privacy. Proc. of the 1999 IEEE Symposium on IEEE*.1-13.

Lee, Y.J. and Mangasarian, O.L. (2001). Ssvm: A smooth support vector machine for classification. *Computational optimization and Applications*. 20(1):5–22

Leonardo, E. (2013). About X.800 and RFC 2828.

Lewis, D.D. and Gale, W.A. (1994). A Sequential Algorithm for Training Text Classifiers SIGIR 94, *Proceedings of Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. Springer-Verlag, London. 3-12.

Lewis, J.P. (2004). Tutorial on Support Vector Machine. CGIT Lab, USC.

Li, Y. , Li, D. ,Cui, W. and Zhang, R.(2011). Research based on OSI model, *3rdInternational Conference on Communication Software and Networks*. IEEE.

Liao, H.J., Lin, C.H.R., Lin, Y.C. and Tung, k.Y. (2013). Intrusion detection system: A comprehensive review, *Journal of Network and Computer Applications*. 36: 16-24.

Lichodzijewski, A. N., Heywood, Z. and Heywood, M.I. (2002). Host-based Intrusion Detection using Self-organising Map. *Proceedings of 2002 IEEE World Congress on Computational Intelligence*, Honolulu, HI. 1714-1719.

Lippmann, R. P., Fried, D.J., Graf, I., Haines, J.W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R.K. and Zissman, M.A. (2000). Evaluating intrusion detection systems, the 1998 darpa off-line intrusion detection evaluation, *DARPA Information Survivability Conference and Exposition*. 2:12-16.

Liul, Y., Cheng, J., Yanl, C., Wul, X. and Chenl, F. (2015). Research on the Matthews Correlation Coefficients Metrics of Personalized Recommendation Algorithm Evaluation, *International Journal of Hybrid Information Technology*. 8(1): 163-172.

Lockhart, A. (2007). Network Security Hacks: Tips and Tools for Protecting Your Privacy, New Delhi, Shroff Publishers and Distributors Pvt. Ltd: 348-349.

Magalhaes, R.M. (2003). Host-Based Intrusion Detection System vs Network-Based Intrusion Detection System (part-1). Windows Security Newsletter.

Mangasarian, O.L. and Musicant, D.R. (1999). Successive over relaxation for Support Vector Machines. *IEEE Transactions on Neural Networks*. 10(5): 1032–1037.

Mangasarian, O.L. and Musicant, D.R. (2001). Lagrangian Support Vector Machines, *Journal of Machine Research*. 1: 161-177.

Manikopoulos, C. and Papavassiliou, S.(2002). Network intrusion and fault detection: A statistical anomaly approach, IEEE Communications Magazine. 40(10):76-82.

Mann, H.B. and Whitney, D.R. (1947). On a test of whether one of two random variables is stochastically larger than the other, *The Annals of Mathematical Statistics*. 50-60.

MATLAB (2015). Neural Network guide. Version 8.50.197613 (R 2015a).

McHugh, J. (2000). Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory, *ACM Transactions on Information and System Security*. 3(4). http:// www.cs.cmu.edu/~maxion/courses/ mchugh00.pdf.

McHugh, J. (2000). Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. *ACM Transactions on Information and System Security*. 3(4):262-294.

McLuhan, M. (1987). Letters of Marshall McLuhan, Oxford University Press. 254.

Mercer, J. (1909). Functions of positive and negative type and their connection with the theory of integral equations. Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character. 415- 446.

Mitchell, T. (1997). Machine Learning, McGraw-Hill Computer science series.

Mohamed, A., Dahl, G. and Hinton, G. (2012). Acoustic modeling using deep belief networks, *IEEE Transactions on Audio, Speech, and Language Processing*. 20(1):14 -22.

Moradi, M. and Zulkernine, M. (2004). A Neural network-based system for intrusion detection and classification of attacks. Natural Sciences and Engineering Research Council of Canada (NSERC). 148-04: 1-6.

Mukkamala, S. and Sung, A.H. (2003). Identifying Significant Features for Network Forensic Analysis Using Artificial Intelligent Techniques, *International Journal of Digital Evidence*. 1(4): 1-17.

Mukkamala, S. and Sung, A.H. (2003a). Artificial intelligent techniques for intrusion detection ,*IEEE International Conference on Systems, Man and Cybernetics*. 2.1266-1271. IEEE.

Mukkamala, S. and Sung, A.H. (2003b). Detecting denial of service attacks using support vector machines. In Fuzzy Systems, 2003. FUZZ'03. *The12th IEEE International Conference on Fuzzy System and Knowledge Discovery*. 2: 1231-1236.

Mukkamala, S. and Sung, A.H. (2003c). Feature selection for intrusion detection with neural networks and support vector machines. Transportation Research Record, *Journal of the Transportation Research Board.* 1822(1): 33-39.

Mukkamala, S., Janoski, G. and Sung, A. (2002). Intrusion detection using neural networks and support vector machines. *Proceedings of the 2002 International Joint Conference* on *Neural Networks*. IJCNN'02. 2: 1702-1707. IEEE.

Mukkamala, S., Janoski, G. and Sung, A. (2003). Intrusion Detection: Support Vector Machines and Neural Networks. 1-9. www.researchgate.net.

Mulay, S.A., Devale, P.R. and Garje, G.V. (2010). Intrusion Detection System using Support Vector Machine and Decision Tree, *International Journal of Computer Applications.* 3(3): 40-43.

Myers, M.(2009). Managing and Trobleshooting Networks, 2nd Ed., New Delhi, McGrawHill.

National Computer Board, Mauritius. (2011). Guideline on Intrusion Detection and Prevention System. Version 1.10. 14-16.

National Information Assurance Glossary (2006). http://jitc.fhu. disa. mil/pki/ documents/ committee_on_national_security_systems_instructions_4009 _june_2006.pdf.

National Institute of Standards and Technology, National Vulnerabilities Database https://web.nvd.nist.gov/view/vuln/statistics.

National Security Agency. (2001). Intrusion Detection System Analyzer Protection Profile, Version 1.1, Fort Meade, National Security Agency. https://www. Common criteria portal. org/files/ppfiles/PP_IDS_ANA_V1.1.pdf.

Nazir, A. (2013). A Comparative Study of Different Artificial Neural Networks Based Intrusion Detection Systems, *International Journal of Scientific and Research Publications*. 3(7): 1-15.

Nettleton, D.F., Puig, A.O. and Fornells, A.(2010). A study of the effect of different types of noise on the precision of supervised learning techniques, Artificial Intelligence Review. 33(4):275-306.

Newman, D., Manalo, K.M. and Tittel, E. (2004), “Intrusion detection overview”, www.informit.com/articles/article.aspx?p¼174342.

NIIT (2004). Information Security: An Overview, New Delhi, Prentice Hall of India Private Limited.

Nimda and Code Red (2009). Dynamic graphs of the Nimda Worm. http:// www. caida. org / dynamic/analysis/security/nimda/.

NIST SP 800-18 (www.csrc.nist.gov/publications/nistpubs).

NIST SP 800-92. Guide to Computer Security Log Management. http://csrc.nist.gov/ publications/nistpubs/.

Norton Cyber Security Insights Report. (2016). Global Comparisons. https://www.Symantec . com/content/dam/symantec/docs/reports/2016-norton-cyber-security-insights-comparisons-us-en.pdf.

Novikov, D.V. (2005). Neural networks to intrusion detection. Thesis. Rochester Institute of Technology. http://scholarworks.rit.edu/theses.

NSL-KDD Data. https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/ NSL-KDD/

Number of internet users worldwide from 2005 to 2017. https://www.statista. com/statistics/273018/number-of-internet-users-worldwide/.

Olusola, A.A., Oladele, A.S. ,andAbosede, D.O.(2010). Analysis of KDD’99 Intrusion Detection Dataset for feature Selection of Relevance Features, *In: Proc. Of the World Congress on Engineering and Computer Science*, San Francisco, USA.1:162-168.

Open Group. (2009). https://en.wikipedia.org/wiki/Threat_ (computer)#cite_note-5.

Osuna E., Freund R., and GirosiF.(1997). Support Vector Machines: Training and Applications. A.I. Memo No. 1602. Artificial Intelligence Laboratory, MIT.

Padhy, N. P. and Simon S.P. (2015). Soft Computing: With Matlab Programming. Canada; Oxford University Press.

Panda Security, Virus, Worms, antivirus and Security Information, http://www.Panda security.com/india/homeusers/security-info/default.aspx?lst=LastVirus&paginaLV=2.

Panda, M. , Das, S. , Patra, M.R. and Abraham, A.(2011). Network Intrusion Detection System: A Machine Learning Approach, *Article in Intelligent Decision Technologies*. 5(4):347-356.

Panko, R. R. (2012). Corporate Computer and Network Security, 2nd ed., New Delhi.

Patel, A. , Qassim, Q. and Wills, C. (2010). A survey of intrusion detection and prevention systems, *Information Management & Computer Security*. 18(4): 277-290.

Patel, R., Thakkar, A. and Ganatra, A. (2012). A Survey and Comparative Analysis of Data Mining Techniques for Network Intrusion Detection Systems, *International Journal of Soft Computing and Engineering (IJSCE)* .ISSN. 2231-2307.

Patil, D.R. and Pattewar, T.M. (2014). A comparative Performance Evaluation of Machine Learning-Based NIDS on Benchmark Datasets. *International Journal of Research in Advent Technology*.2 (2): 101-106.

Patnaik, S. (2001). First Text Book on Information Technology, New Delhi, Dhanpat Ray &Co. 363.

Peddabachigari, S., Abraham, A., Grosan, C., & Thomas, J. (2007). Modeling intrusion detection system using hybrid intelligent systems, *Journal of network and computer applications*. 30(1):114-132.

Pervez, S., Ahmad, I., Akram, A., Swati, S.U. (2006). A Comparative Analysis of Artificial Neural Network Technologies in Intrusion Detection Systems, *In Proceedings of the 6th WSEAS International Conference on Multimedia, Internet & Video Technologies*. Lisbon, Portugal. 84-89.

Philips, C. and Swiler, L.P. (1998). A graph-based system for network-vulnerability analysis, *Proceedings of the 1998 workshop on New security paradigms*.71-79.

Porras, A. and Neumann, P.G. (1997). EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances, Proc. 20th NISSC, 353- 365, http://www.sdl. sri.com/ papers / emerald-position1/) To check (Performance enhancement thesis).

Powers, D.M.W. (2007). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness& Correlation. Technical Report SIE-07-001, School of Informatics and Engineering. 1-24.

Powers, D.M.W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness& Correlation, *Journal of Machine Learning Technologies* 2(1): 37–63.

Quinlan, J.R. (1993). *C.4.5: programs for machine learning*, San Mate:Morgan Kaufmann Publishers, San Francisco, CA, USA.

Race, R. G. (2000). Intrusion Detection, USA, McMillan Technical Processing.

Rajagopalan, M., Hiltunen, M.A., Jim, T. and Schlichting, R.D. (2006). System Call Monitoring Using Authenticated System Calls, *IEEE Transactions on in dependable and Secure Computing*. 3(3): 216-229.

Ramachandran, V. and Nandi, S.(2005). Detecting ARP Spoofing: An Active Technique, *International Conference on Information Systems Security*. 239-250. Lecture Notes in Computer Science. 3803. Springer, Berlin, Heidelberg.

Ranshous, S., Shen, S., Koutra, D., Faloutsos, C. and Samatova, N.F. (2014). Anomaly Detection in Dynamic Networks: A Survey. *WIREs Computational Statistics*. 7**:** 223-247.

Raychaudhuri, K. , Manoj Kumar,M. and Sanjana Bhanu, S.(2017). A Comparative Study and Performance Analysis of Classification Techniques: Support Vector Machine, Neural Networks and Decision Trees, *Springer Nature Singapore Pte Ltd*. M. Singh et al. (Eds.): ICACDS 2016, CCIS 721, pp. 13–21, 2017.DOI: 10.1007/978-981-10-5427-3_2.

Reddy, E.K. (2013). Neural Networks for Intrusion Detection and Its Applications, *Proceedings of the World Congress on Engineering*. Vol. II.WCE, London, U.K.

Reddy, R. R. , Kavya, B. and Ramadevi,Y. (2014). A survey on SVM for Intrusion Detection, *International Journal of Computer Application*. 98(19): 38-43.

Rejchrt, J. (2014), Network Anomaly Detection – Survey Evaluation, https:// labs. ripe. net/ Members/jan_rejchrt/network-anomaly-detection-2013-survey-evaluation.

Revathi, S. and Malathi, A. (2013). A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection**,** *International Journal of Engineering Research & Technology* (IJERT). 2(12): 1848-1853.

Rivas, V. M., Merelo, J. J., Castillo, P. A., Arenas, M. G. and Castellano, J. G. (2004). Evolving RBF neural networks for time-series forecasting with EvRBF, *Information Sciences*. 165(3): 207-220.

Robertson, W., Vigna, G., Kruegel, C. and Kemmerer, R.A. (2006). Using Generalization and Characterization Techniques in the Anomaly-based Detection of Web Attacks. https://www.cs.ucsb.edu/~vigna/ publications/ 2006_ robertson_ vigna_ kruegel_ kemmerer_NDSS.pdf.

Rojas, R. and Feldman, J. (1996). Neural Networks: A Systematic Introduction . Springer-Verlag, Berlin, New-York.

Roohi, F. (2013). Artificial Neural Network Approach to Clustering. *The International Journal of Engineering and Science*. 2(3): 33-38.

Roque, A. (n.d). Architecture of Intrusion Detection using Intelligent Agents, users. cis.fiu. edu/~aleroque/intru_det.doc.

Rossius, R., Zenker, G., Ittner, A., and Dilger, W. (1998). A short note about the application of polynomial kernels with fractional degree in Support Vector Learning, *In Machine Learning: ECML-98*.143-148. Springer. Berlin. Heidelberg.

Rout, N., Mishra, D. and Mallick, M. K. (2017). Analysing the Multi-class Imbalanced Datasets using Boosting Methods and Relevant Information,*International Journal of Computer Technology and Applications.* 10(9): 907-921.

Ryan A. R. , Frederick, C. and Lepes, D. (1997). Intrinsic Motivation and Exercise Adherence,*International journal of sport psychology*. 28(4):335-354.

Sadoddin, R. and Ghorbani, A.A. (2007). A comparative study of unsupervised machine learning and data mining techniques for intrusion detection, *In Machine Learning and Data Mining in PatternRecognition*. Springer. Berlin. Heidelberg.404-418

Sameh, N., Gayar, N.El and Nashwa, A.B. (2008) Misfeasor Classification and Detection Models using Machine Learning Techniques, *Journal of Information Assurance and Security*. 4(10):1.

Sammany, M. ,Sharawi, M. , Beltagy, E. M. and Saroit, I. (2007). Artificial Neural Networks Architecture for Intrusion Detection Systems and Classification of Attacks ,*In: Proc. 5th Int. Conf. INFO, Cairo University*.

Sayer, A. A. ,Pawar, S. N. and Mane, V. (2014). A Review of Intrusion Detection System in Computer Network, *International Journal of Computer Science and Mobile Computing*. 3(2): 700-703.

Sayyadi, H., Hurst, M. and Maykov, A. (2009). Event Detection and Tracking in Social Streams, *Proceedings of the Third International ICWSM Conference*. https:// www.cs. umd.edu/~sayyadi/files/papers/9-sayyadi-EventDetection_KeyGraph_ICWSM09 .pdf.

Scarfone, K. and Mell, P. (2007). Computer Security, Guide to Intrusion Detection and Prevention Systems (IDPS), Recommendations of the National Institute of Standards and Technology. http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf.

Scarfone, K. and Mell, P. (2010).Intrusion Detection and Prevention Systems. In: Stavroulakis P., Stamp M. (eds) Handbook of Information and Communication Security. Springer, Berlin, Heidelberg.

Schaffer, G. (2007). Building a cheap and powerful intrusion detection system, *Computer world.* http://www.computerworld.com/article/2541227.

Scholkopf, B., Burges, C.J.C. and Smola, A.J. (1999). Advances in kernel methods: support vector learning. MIT press.

Schuster, A. (2008). Robust Artificial Neural Network Architectures, *World Academy of Science, Engineering and Technology*. 2. https://www.waset.org/Publications/.

Security threat report. (2014). https://www.sophos.com/en-us/medialibrary/PDFs/other/ sophos-security-threat-report-2014.pdf.

Sengupta, N., Sen, J., Sil, J. and Saha, M. (2013). Designing of online intrusion detection system using rough set theory and Q-learning algorithm, *Neurocomputing*. 111:161-168.

Shah, B and Trivedi, B.H. (2012). Artificial Neural Network based Intrusion Detection System: A Survey, *International Journal of Computer Applications*. 39(6): 13-18.

Sharma, N. and Mukherjee, S.(2012). A Novel Multi-Classifier Layered Approach to Improve Minority Attack Detection in Intrusion Detection System, *Procedia Technology*. *2nd International Conference on Communication, Computing &Security* ,ICCCS. 6: 913-921.

Shin, M. and Park, C. (2000). A Radial Basis Function Approach to Pattern Recognisation and its Application. *Electronics and Telecommunications Research Institute Journal*, 22(2):1-10.

Shirey, R. (2000), Internet Security Glossary. RFC Editor. ACM Digital Library.

Shrivastava, A., Baghel, M. and Gupta, H. (2013). A Review of Intrusion Detection by Soft Computing and Data Mining Approach, *International Journal of Advanced Computer Research*. 3(12):224-228

Shum, J. and Malki, H.A. (2008). Network Intrusion Detection System using Neural Network. http://ieeexplore.ieee.org/iel5/4666791/4667376/04667434. pdf.

Simakov, S. (2005). Introduction to MATLAB Graphical User Interfaces. Maritime Operations Division Defence Science and Technology Organisation. DSTO–GD–0442.1-46

Simpson, M. T. (2003). Ethical Hacking and Network Defense, Australia, Course Technology Cengage Learning. 335.

Sinclair, L., Pierce. and Matzner, S. (1999). An Application of Machine Learning to Network Intrusion Detection, *Proceedings of 15th Annual Computer Security Application Conference,* (ACSAC'99), Phoenix, AZ: 371-377.

Singh, P.K., Vatsa, A.K., Sharma, R. And Tyagi, P.(2012).Taxonomy Based Intrusion Attacks and Detection Management Scheme in Peer to Peer Network, *International Journal of Network Security & Its Applications* (IJNSA). 4(5): 167-179.

Singh, R. ,Kumar, H. and .Singla, R.K.(2015). An intrusion detection system using network traffic profiling and online sequential extreme learning machine,*Expert Systems With Applications*. 42: 8609–8624.

Sivanandam, S. N., Sumathi, S., and Deepa, S.N. (2006). Introduction to Neural Networks using MATLAB 6.0. New Delhi: McGraw Hill Education (India) Pvt. Ltd.

Smotroff, I. G., Friedman, D. H. and Connolly, D. (1991). Self Organizing Modular Neural Networks, *International Joint Conference on Neural Networks*. 2:187-192. DOI 10.1109/IJCNN.1991.155336

Snapp, S.R., Brentano, J., Dias, G.V., Gaon, T.L., Heberlein, L.T., Ho, C.L., Levitt, K.N., Mukherjee, B., Smahal, S.E., Grance, T., Teal, D.M. and Mansur, D. (1991). DIDS (Distributed Intrusion Detection System- Motivation, Architecture and an Early prototype, *In. Proceedings of 14th National Computer Security Conference*. 167-176.

Sodiya, A.S., Ojesanmi, O.A. , Akinola, O.C. and Aborisade, O. (2014). Neural Network based Intrusion Detection Systems, *International Journal of Computer Applications.* 106 (18): 19-24.

Somer, R. (2010). Viable Network Intrusion Detection : Trade-offs in High Performance Environments, VDM Verlag Dr. Muller, Germany. 9-11.

Sommestad, T. and Hunstad, A. (2013). Intrusion detection and the role of the system administrator, *Information Management & Computer Security*. 21(1):30 – 40.

Sorensen, S. (2006). Intrusion Detection and Prevention Protecting Your Network From Attacks, Sunnyvale Juniper Networks, Inc., http://www.cstl.com/Products/ Juniper/ Juniper-IDP-Solution/WhitePaper/ Juniper-IDPWhitePaper.pdf

Spafford, E.H. (1988). The Internet Worm Program: An Analysis. Computer Science Technical Reports. Paper 702. http://docs.lib.purdue.edu/cstech/702.

Srinoy, S., Kurutach, W., Chimphlee, W. and Chimphlee, S. (2007). Network Anomaly Detection using Soft Computing, *International Journal of Computer, Information, Systems and Control Engineering.* 1(9): 2864-2868.

Stakhanova, N., Basu, S. and Wong J.S. (2006). A Taxonomy of Intrusion Response Systems, *Computer Science Technical Reports.* Paper 210. 1-18.

Stallings, W. (2011). Network Security Essentials: Applications and Standards, Fourth ed., Chennai, Pearson.

Stallings, W. and Brown, L. (2007). Computer Security: Principles and Practice. 1st. Ed., New Delhi, Prentice Hall.

Stallings, W. and Brown, L. (2010). Computer Security: Principles and Practice, New Delhi, Person Education, Inc.

Stolfo, S.J., Fan, W. , Lee, W. , Prodromidis, A. and Chan, P.K. (2000). Cost based Modelling for fraud and intrusion detection: Results from the jam project, *In: Proc. of DARPA Information survivability Conference and exposition, DISCEX*. 2:130-144.

Sumathi, S. and Paneerselvam, S. (2010). Computational intelligence paradigms: Theory and Applications using MATLAB. Boca Raton, FL, USA: CRC Press.

Sung, A.H. and Mukkamala, S. (2003). Identifying important features for intrusion detection using support vector machines and neural networks, *In: Proc. of Applications and Internet*. IEEE : 209-216.

Sutskever, I. (2013). Training Recurrent Neural Networks. Ph.D. Thesis. Graduate Department of Computer Science University of Toronto.

Sutskever, I., Martens, J. and Hinton, G. (2011). Generating text with recurrent neural networks, *In:Proceedings of the 28th International Conference on Machine Learning* (ICML-11), ICML '11 :1017-1024.

Symeonidis, K. (2000). Hand Gesture Recognition Using Neural Networks. http:// personal . ee.surrey.ac.uk/Personal/T.Windeatt/msc_projects/symeonidis/ Final% 20 Project.pdf.

Tan, Y. and Wang, J. (2004). A support vector machine with a hybrid kernel and minimal Vapnik-Chervonenkis dimension, *IEEE Transactions on knowledge and data engineering*. 16(4): 385-395.

Tanenbaum, A. S. (2003). Computer Networks, *4th ed.* New Delhi, Dorling Kindersley (India) Pvt. Ltd.

Tang, H. and Ghorbani, A.A. (2003) . Accent Classification Using Support Vector Machine and Hidden Markov Model, *Advances in Artificial Intelligence*. 629-631.https://pdfs.semanticscholar.org / dee5/eadd08480 737277585c44485186fedc0c 946. pdf.

Tavallaee, M., Baghari, E. , Lu, W. and Ghorbani, A.A.(2009). A detailed analysis of the KDD CUP 99 datasets, *In. Proc. 2nd IEEE Symposium on Computational Intelligence in Security and Defence Applications*. 53-58.

Theodorios, S and Koutroumbas, K. (1999). Pattern Recognisation, Cambridge, Academic Press.

Thomas, C. (2009). Performance Enhancement of Intrusion Detection Systems using Advances in Sensor Fusion. Ph.D. Thesis. Indian Institute of Science, Bangalore.

Tian, S., Mu, S., and Yin, C. (2007). Sequence-similarity Kernels for SVMs to Detect Anomalies in System Calls, *Neurocomputing*. 70 (4-6): 859–866.

Ting, K. M. (2011). *Encyclopedia of machine learning*. Springer. ISBN 978-0-387-30164-8.

Trends in Attack Sophistication and Intruder Knowledge, sophisticated command and control Source: CERT. DDoS attacks. Intruder Knowledge, http://www.letu.edu/people/ jaytevis/Network-Security/Stallings-Figures/Ch-01-Introduction/f1-2. pdf.

Triguero, I., Gonzalez, S., Moyano, J.M., Garcia, S., Alcala-Fdez, J., Luengo, J., Fernandez, A., Jesus, M. J. Del., Sanchez, L. and Herrera, F. (2017). KEEL 3.0: An Open Source Software for Multi-Stage Analysis in Data Mining, *International Journal of Computational Intelligence Systems*.10(1): 1238-1249.

Tsang, I.W., Kwok, J.T. and Cheung, P.M. (2005). Core Vector Machines: Fast SVM Training on Very Large Data Sets, *Journal of Machine Learning Research*. 6: 363-392.

Vamsidhar, T., Reddyboina, A. and Rayala, V. (2012). Intrusion Detection System for Web Application with attack classification, *Journal of Global Research in Computer Science.* 3(12): 44-50.

Vapnik, V. (1995). The Nature of Statistical Learning Theory. Springer-Verlag , New York.

Vasan, K.K. and Surendiran, B.(2016).Dimensionality reduction using Principal Component Analysis for network intrusion detection, *Perspectives in Science*. 8: 510-512.

Venkatesan, R., Lakkavalli, S. (2001). TCP/IP Vulnerabilities.1-17. http:// cs.Ucsb.edu/~koc/ ns /projects/00Reports/LV.pdf.

Vinchurkar, D. P. and Reshamwala, A. (2012). A Review of Intrusion Detection System Using Neural Network and Machine Learning Technique, *International Journal of Engineering Science and Innovative Technology* (IJESIT). 1(2**)**: 54-63

Vinueza , A. and Grudic, G. (2004). Unsupervised Outlier Detection and Semi-Supervised Learning,*Computer Science Technical Reports*. 914.

Vipin Kumar, Chauhan, H., Panwar, D. (2013). K-Means Clustering Approach to Analyze NSL-KDD Intrusion Detection Dataset, *International Journal of Soft Computing and Engineering (IJSCE).* 3(4): 1-4.

Wagner, D. and Soto, P. (2002). Mimicry Attacks on Host Based Intrusion Detection Systems, *Proceedings of the 9th ACM Conference on Computer and Communication Security*. https://www.eecs.berkeley.edu/~daw/papers/ mimicry.pdf.

Wang, G. , Hao, J., Ma, J. and L. Huang, L.(2010). A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering. 37(9): 6225-6232.

Wang, J. (2009). Computer Network Security. Theory and Practice, Beijing, Higher Education Press.

Wang, J. (2011). Research on Software Buffer Overflow Flaw Model and Test Technology,*International Conference on Web Information Systems and Mining.*Springer, Berlin, Heidelber. 238:192-202.

Wang, J. and Kissel, Z.A. (2015). Introduction to Network Security: Theory and Practice. 2nd ed., USA, Wiley.

Wang, J. and Yu, Y. (2013). Research on Hybrid Neural Network in Intrusion Detection System.*World Academy of Science, Engineering and Technology*. 7(4): 481-485.

Whitman, M.E. and Mattord, H.J. (2005). Principles of Information Security. Thomson Course Technology, Boston, MA.

Witten, I. , Frank, H. E. and Hall, M. A. (2011). Data Mining: Practical Machine Learning Tools and Techniques (3rd ed.). Morgan Kaufmann Publishers Inc.

Wu, X. , Kumar, V. , Ross Quinlan, J. , Ghosh, J. , Yang, Q. , Motoda, H. , McLachlan, G. J. , Ng, A., Liu, B. ,Yu, P.S., Zhou, Z.H. ,Steinbach, M. ,Hand, D.J., Steinberg, D. (2008). Top 10 algorithms in data mining, *Knowledge and information systems*.14: 1-37.

Wua, S.Y. and Yen, E. (2009). Data mining-based intrusion detectors, *Expert Systems with Applications*. 36: 5605–5612.

Yao,Y.Y., Wong, S.K.M. and Lin, T.Y. (1997). A Review of Rough set Model. *In*.Rough Sets and Data Mining Analysis of Imprecise Data. Lin, T.Y., Cercone, N. (Eds.). The Netherlands, Kluwer Academic Publishers**.**

Yegnanarayan, B. (2006). Artificial Neural Networks, New Delhi, Prentice Hall of India Pvt. Ltd. 24.

Yekkehkhany, B. , Safari1, A., Homayouni, S., and Hasanlou, M. (2014).. A Comparison Study of Different Kernel Functions for SVM-based Classification of Multi-temporal Polarimetry SAR Data ,*The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol.XL-2/W3. *The 1st ISPRS International Conference on Geospatial Information Research*. 15-17. Tehran, Iran.

Yuan, Y.X. and Sun, W. (1997). Optimization Theories and Methods., Beijing, Science Press.

Zahariev, A. (2011). Graphical User Interface for Intrusion Detection in Telecommunications Networks. http://www.futureinternet.fi/ publications /zahariev2011.pdf.

Zainal, A., Maarof, M.A., Shamsudin. S.M. (2006). Research Issues in Adaptive Intrusion Detection, *In Proceedings of the 2nd Postgraduate Annual Research Seminar (PARS'06*). 138-146.

Zanero, S. (2007). Flaws and Frauds in the evaluation of IDS/IPS technologies, *In: Proc. Of Forum of Incident Response and Security Teams (First 2007)*:1-18.anni, L., Serafini, T. and Zanghirati, G. (2006). Parallel Software for Training Large Scale Support Vector Machines on Multiprocessor Systems, *Journal of Machine Learning Research Archive*. 7: 1467-1492.

Zeltser, L. (2011). Computer Attacks Defences. In. Technical Guide on Emerging Threats. Newton, Tech Target: 1-8.

Zhang, R. , Wang. Y. and Liu, Y. (2014). Research on File System Event Monitoring in Android Platform, *Applied Mechanics and Materials*. 443. 494-498.

Zhang, X., Gu, C. and Lin ,J. (2006b). Support vector machines for anomaly detection ,*In: Proc. of the 6th World Congress on Intelligent Control and Automation*: 2594-2598.

Zhao, J. and Li, W. (2012). Improvement Intrusion Detection Based on Support Vector Machine. Springer-Verlag Berlin Heidelberg.

Zhivich, M., Leek, T. and Lippmann, R. (2005). Dynamic Buffer Overflow Detection, https://www.cs.umd.edu/~pugh/BugWorkshop05/papers/61-zhivich.pdf.

Zhou, J. and Gollmann, D. (1997). Evidence and non-repudiation, *Journal of Network and Computer Applications.* 20: 267–281

Zhou, W., Jia, W. ,Wen, S., Xiang, Y. And Zhou, W. (2013). Detection and defense of application-layer DDoS attacks in backbone web traffic, *Future Generation Computer Systems.*

Zhou, Z. (2004). Rule Extraction using Neural network or for Neural Networks, *Journal of Computer Science and Technology*. 19(2): 249-253.

Zhu, W., Zeng, N. and Wang, N. (2010). Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS implementations. NESUG proc. healthcare and life sciences, Baltimore, Maryland: 1-9.

Zien, A., Ratsch, G., Mika, S., Scholkopf, B., Lemmen, C., Smola, A., Lengauer, T. and Muller, K. R. (2000). Engineering support vector machine kernels that recognize translation initiation sites, *Bioinformatics*. 16(9): 799-807.

Zou, Y. and Chakrabarty, K. (2003). Sensor Deployment and Target Localization Based on Virtual Forces. IEEE INFOCOM.

BIBLIOGRAPHY

BRIEF BIO-DATA OF THE CANDIDATE

1.	Name:	**Aishwarya Mishra**
2.	Designation:	Research Scholar
3.	Complete Postal Address:	Department of Mathematics and Computer Science Mizoram University, Aizawl- 796004
4.	Permanent Address:	**C/o. Prof. Rabinarayan Mishra** At- Mohantypara, Sambalpur- 768 001 (Odisha) ☎ 9337974463(M), aishwaryamca@rediffmail.com, aishwaryamishra1987@gmail.com
5.	Date of Birth:	**24th May 1987**
10.	Educational Qualifications:	

Examination	**University / Board**	**Year of passing**	**Marks obtained**	**SGPA/ CGPA(% of marks)**	**Class/ Division**	**Subjects**
H.S.C.	B. S. E, Odisha	2002	492/750	65.6	1st	Eng., Oriya, Sansk., Math, Science, etc
+2 Arts	C.H.S.E.,Odisha	2004	467/900	51.8	2nd	Eng., Oriya, Sans, Logic , Math., & Statistics
+3 Arts	Sambalpur University	2007	1180/ 1800	65.55	1st	Statistics Hons, Geography, Pol.Sc.
			660/800	82.5		
MCA	Biju Patnaik University of Technology (BPUT), Odisha	2007-2010	1st Sem.	7.70	1st	Comm Eng., Engin Eco. & Costing, C, MALP, Dis. Math.
			2nd Sem.	7.69	CGPA- 8.07%	Num. Method, Bus. Comm. in Eng, Data Structure

						using C, Comp. syst. Art., C++
			3rd Sem.	7.19		Fin & Mang. Accounting, RDBMS, OS, MIS, ADA
			4th Sem.	8.00		Qunt. Tech-I, Core Java, Comp. Graphics, SE&OOAD, HRM
			5th Sem.	8.20		Comp. Networks, Enterprise Adv. Java, Qunt. Tech-II, Compt. Security (Elect), IT
			6th Sem.	10		Project on Production Management of IFCAL
M.Tech in Computer science	Sambalpur University Institute of Information Technology (SUIIT)	2010-12	1st Sem.	5.82	CPI 6.05%	Math. Foundation, DSA, ACOA, CN, SE
			2nd. Sem	6.33		AI, AOS, CNS, WSN, ML
			3rd Sem.	6.00		Dissertation-I
			4th Sem.	6.00		Dissertation-II
		2010-12	Course Completed		5.75% (CPI) (2nd)	
Ph.D	Mizoram University, Aizawl	2015	2020			Computer Science

11. **Additional Qualifications:**

i.	Passed Beginners Exposure to Software Technology in 2002 from NICE, Sambalpur.
ii.	Passed Diploma in Computer Application in 2005 with 1st Div. from NICE, Sambalpur.
iii.	Completed Interface Certified Java Programmer for the Java 2 Platform 1.4/1.5 of Sun Microsystems from Interface Software, Bhubaneswar from January 2009 to April 2009.

12. **Teaching Experience**: July 2012- December, 2013 at G.M (Auto) College, Sambalpur in MCA Department.

13. **Credit**

i Declared as Best Graduate (First class First) in Arts from G.M College (Auto.), Sambalpur in 2007

ii Certificate of Merit issued by Bharat Vikash Parisad

14. **Seminar Presentation**

Sl.No.	**Authors**	**Title of the seminar/ paper**	**Year**	**Remarks**
1	A Mishra	Network Computing System	2008	Presented
2	A Mishra	Wireless Networked digital devices: A new paradigm for computing and communication	2009	Presented
3	R N Mishra & A Mishra	Magnitude and Activities of Knowledge Management and its application in 21st century Libraries.	2011	Accepted for Poster presentation in International CALIBRE-2011 (INFLIBNET) at , Goa University, Goa
4	A Mishra & R N Mishra	E-Learning in Computer Mediated Communication System: Issues and Challenges for Future Librarianship	25-26 Nov., 2016	Presented paper in International Conference on Marching Beyond Libraries: The Role of Social Media and Networking at KIIT University Bhubaneswar

Date: 25.2.20

(Aishwarya Mishra)

Place: Aizawl

PARTICULARS OF THE CANDIDATE

Name of the Candidate:	Mrs. Aishwarya Mishra
Degree:	Doctor of Philosophy
Department:	Mathematics and Computer Science
	Mizoram University, Aizawl
Title of the Thesis:	**INTRUSION DETECTION SYSTEM USING ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINE: A COMPARATIVE STUDY**
Date of Admission:	04.09.2014
Approval of the Research Proposal	
DRC	**24.02.2020**
BOS	**06.05.2015**
School Board	**11.05.2015**
MZU Registration No.	**153 of 2015**
Ph.D Registration Number:	**MZU/Ph.D./741 of 11.05.2015**
Date Extension, if any	**No**

Aishwarya Mishra

HEAD

Department of Mathematics and Computer Science

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
MIZORAM UNIVERSITY

Dr. Jay Prakash Singh
Associate Professor & Head

Aizawl- 796004, Mizoram
India
Phone- +91 8974134152 (M)
E-mail: jpsmaths@gmail.com

CERTIFICATE OF HEAD OF DEPARTMENT

This is to certify that Ms. Aishwarya Mishra, a Ph.D. Scholar, Date of Admission 04.09.2014 and Registration No. MZU/Ph.D./741 of 11.05.2015 has worked in the thesis entitled, **“Intrusion Detection System Using Artificial Neural Network and Support Vector Machine: A Comparative Study**”. She has fulfilled all the criteria prescribed by the UGC (Minimum Standard and Procedure Governing Ph.D. Regulation). She has fulfilled the mandatory publication and presentation (copy enclosed) and completed M.Phil./Ph.D. Course work. It is also certified that the scholar has been admitted to the Department through an entrance test followed by an interview as per Clause 9 (i) and (ii) of the UGC Regulation, 2009.

(Dr. Jay Prakash Singh)
Head
Department of Mathematics and
Computer Science
Mizoram University