A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection

Preeti Mishra, *Member, IEEE*, Vijay Varadharajan, *Senior Member, IEEE*, Uday Tupakula, *Member, IEEE*, and Emmanuel S. Pilli, *Senior Member, IEEE*

***Abstract*—Intrusion detection is one of the important security problems in todays cyber world. A significant number of techniques have been developed which are based on machine learning approaches. However, they are not very successful in identifying all types of intrusions. In this paper, a detailed investigation and analysis of various machine learning techniques have been carried out for finding the cause of problems associated with various machine learning techniques in detecting intrusive activities. Attack classification and mapping of the attack features is provided corresponding to each attack. Issues which are related to detecting low-frequency attacks using network attack dataset are also discussed and viable methods are suggested for improvement. Machine learning techniques have been analyzed and compared in terms of their detection capability for detecting the various category of attacks. Limitations associated with each category of them are also discussed. Various data mining tools for machine learning have also been included in the paper. At the end, future directions are provided for attack detection using machine learning techniques.**

***Index Terms*—Machine learning, intrusion, attacks, security.**

I. Introduction

HACKING incidents are increasing day by day as technology rolls out. A large number of hacking incidents are reported by companies each year. Distributed Denial of Service (DDoS) attack was launched against Estonian websites in 2007, allegedly by Russia [1]. On June 17, 2008, Amazon [2] started receiving some authenticated request from multiple users in one of its location. The requests began to increase significantly causing the servers slow down. On Jan 2013, European Network and Information Security Agency (ENISA) [3] reported that Dropbox was attacked by DDoS and suffered a substantial loss of service for more than 15 hours affecting all users across the globe. Facebook [4] was hit by suspected distributed denial of service attack on Sept 28, 2014. Panjwani *et al.* [5] reported that some form of network scanning activity precedes 50% of the attacks against cyber systems. Attackers are not only launching flooding and probing attacks but also spreading malware files in the form of virus, worm, spams to exploit the vulnerabilities present in existing software, causing a threat to the sensitive information of users stored on machines. Cisco Annual Security report mentioned [6] that spam related to the Boston Marathon bombing comprised 40% of all spam messages delivered worldwide on April 17, 2013. On a recent survey done by Cisco in 2017 [7], Trojan was classified as one of the top five malware which is used to gain initial access to the user's computers and organizational networks. Hence, security in such a complex technological environment is a big challenge and needs to be tackled intelligently.

Researchers have considered a different category of attacks for intrusion detection. For example, Denial of Service (DoS) attacks (Bandwidth and Resource Depletion), Scanning attacks (Probe) and Remote to Local (R2L) attacks and User to Root (U2R) attacks which are based on KDD'99 dataset [12]. A recent attack dataset (UNSW-NB [13]), classifies attacks into nine categories: Fuzzer, Analysis, Reconnaissance, ShellCode, Worm, Generic, DoS, Exploit and Generic. All these attacks have been discussed in detail in Section III.

Current security solutions include the use of middle-boxes such as Firewall, Antivirus and Intrusion Detection Systems (IDS). A firewall controls traffic that enters or leaves a network based on source or destination address. It alters the traffic according to the firewall rules. Firewalls are also limited to the amount of state available and their knowledge of the hosts receiving the content. An IDS is a type of security tool that monitors network traffic and scans the system for suspicious activities and alerts the system or network administrator [14]. It is the main focus of concern in this paper.

IDS are mainly two types: Host based and Network based. A Host based Intrusion Detection System (HIDS) [15] monitors individual host or device and sends alerts to the user if suspicious activities such as modifying or deleting a system file, unwanted sequence of system calls, unwanted configuration changes are detected. A Network based Intrusion Detection System (NIDS) [16] is usually placed at network points such as a gateway and routers to check for intrusions in the network traffic.

Manuscript received June 21, 2017; revised November 27, 2017 and April 2, 2018; accepted May 22, 2018. Date of publication June 15, 2018; date of current version February 22, 2019. *(Corresponding author: Emmanuel S. Pilli.)*

P. Mishra was with MNIT, Jaipur 302017, India. She is now with the Department of Computer Science and Engineering, Graphic Era (Deemed) University, Dehradun 248002, India (e-mail: dr.preetimishranit@gmail.com).

V. Varadharajan and U. Tupakula are with the Faculty of Engineering and Built Environment and Advanced Cyber Security Research Centre, University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: vijay.varadharajan@newcastle.edu.au; uday.tupakula@newcastle.edu.au).

E. S. Pilli is with the Department of Computer Science and Engineering, Malaviya National Institute of Technology, Jaipur 302017, India (e-mail: espilli.cse@mnit.ac.in).

Digital Object Identifier 10.1109/COMST.2018.2847722

TABLE I
DIFFERENCE BETWEEN MISUSE DETECTION AND ANOMALY DETECTION

Misuse Detection	Anomaly Detection
It models the existing well known attack patterns/signatures to detect malicious activity. A match of incoming pattern with existing attack profiles is declared as suspicious.	It uses the established normal behavior profile of the system. A mismatch of incoming pattern with the existing normal profile is declared as suspicious.
'Signature-matching' is very popular misuse detection approach, having commercial success.	'Statistical learning' is very popular anomaly detection approach and researchers are still working in this direction.
Supervised machine learning approaches are well suited for misuse detection such as DT, NB, BP-ANN.	Semi or unsupervised machine learning approaches are well suited for anomaly detection such as Clustering, SOM-ANN, One-class SVM etc.
Unable to detect the unknown attacks.	Good for detecting the unknown attacks.
Low-occurrences of false positives	High-occurrences of false positives.
Very good accuracy for detecting known attacks.	Provides good accuracy for unknown attacks.
Ch allege lies in maintaining a up-to-date database of all known attack signatures.	Challenge lies in differentiating the attack and evolving normal behavior.
Exp. SNORT [8], Suricata [9]	Exp. IDES [10], MINDS [11]

At high-level, the detection mechanism used by these IDSes are of three types: misuse detection, anomaly detection, and hybrid detection. In misuse detection approach, IDS maintains a set of the knowledge base (rules) for detecting the known attack types. Misuse detection techniques can be broadly classified into Knowledge based and machine learning based techniques. In the knowledge based technique, network traffic or host audit data (such as system call traces) are compared against predefined rules or attack patterns. Knowledge based techniques can be categorized into three types: (i) Signature matching (ii) State transition analysis and (iii) Rule based expert systems [17].

Signature matching based misuse detection techniques scan the incoming packets against fixed patterns. If any of the patterns match with the packet header, the packet is flagged as anomalous. State transition analysis based approaches, maintain a state transition model of the system for the known suspicious patterns. Different branches of the model lead to a final compromised state of the machine. The rule based expert systems maintain a database of rules for different intrusive scenarios. The knowledge based IDS requires regular maintenance of knowledge database in a dynamic manner and can fail to detect variants of attacks. Misuse detection can also be performed using supervised machine learning algorithms such as Back Propagation Artificial Neural Network (BP-ANN) [18], Decision Tree (DT) C4.5 [19] and Multi-class Support Vector Machine (SVM) [20].

Machine learning based IDS provides a learning based system to discover classes of attacks based on learned normal and attack behavior. The goal of machine learning based IDS (based on supervised learning algorithms) is to generate a general representation of known attacks. Misuse detection techniques fail to detect unknown attacks. However, these techniques provide good detection accuracy for detecting well-known attacks. These type of IDSes also require the regular maintenance of the signature database which increases the overhead of user.

Misuse detection based IDS particularly signature based are very popular and have got commercial success. The pros and cons associated with these approaches are shown in Table I. These IDSes maintain a database of known attack signatures. An attack signature describes the characteristics of an attack. It can be in the form of a code script, a sequence of system call patterns or a behavioral profile, etc. IDS stores the attack signatures in a certain format. Let us consider TCP-ping attack for illustrating the signature based misuse detection system (particularly SNORT [8]). If an attacker wants to know, if a machine is active or not, he/she scans the machine. An attacker sends ICMP ping packets. If the machine is set to not to respond for ICMP ECHO REQUEST ping packets, an attacker may use the nmap tool to send the TCP ping packets to port 80 with ACK flag set with sequence number 0. The characteristics of this attack is that flag is set to 'A' value and acknowledge set to 0 value [21]. As such packets are not acceptable at the victim side; on receiving the packets, RST packet is sent to attacker's machine which signals machine is alive. The rule for detecting TCP ping attack, targeted against victim machine residing in the network with IP 192.168.1.0/24 is as follows:

alert TCP any ->192.168.1.0/24 any,(flags: A;ack: 0; msg: "TCP ping detected";)

The major limitations with signature based IDS is that it requires the regular update of the system for adding signature rules for up-to-date attacks. It generates more false alarms for the new evolving attacks whose signatures are not defined. Later, anomaly detection approaches are used for detecting intrusions.

Anomaly detection based IDSes are based on the hypothesis that attacker's behavior differs from normal user's behavior [22]. It helps in detecting the evolving attacks. Anomaly based IDSes model the normal behavior of the system and keep on updating it over a duration of time. For example, each network connection is identified by a set of features such as protocol, service, number of login attempts, packets per flow, bytes per flow, source address, destination address, source port, destination port, etc. The behavioral statistics of these features are recorded over a period. Any abnormal deviation in the feature values for any connection flow will be marked as anomalous by the anomaly detection engine. Anomaly detection techniques are widely categorized into three types: Statistical techniques, Machine learning based techniques and Finite state machine (FSM) based techniques [23]. A finite state machine (FSM) produces a behavioral model which is composed of states, transitions, and actions. Kumar *et al.* [24] have proposed an IDS which makes use of Hidden Markov Model to model the transitions of user behavior over a longer span of time. Anomaly detection can also be performed using semi-supervised and unsupervised

machine learning algorithms such as Self Organizing Map (SOM) Neural Network [25], clustering algorithms [26] and One class Support Vector Machine (SVM) [27]. Machine learning based IDS for anomaly detection provide a learning based system to discover zero-day attacks. A Zero-day attack refers to exploitation of a vulnerability that has not been known earlier. However, these techniques suffer from high-false positives because of their limitations in differentiating attack behavior and evolving normal behavior. The difference between misuse detection and anomaly detection approach is shown in Table I. Hybrid detection approaches integrate misuse and anomaly detection approach for detecting attacks. The details of these approaches with the example of existing literature is presented in Section V. In general, some of the advantages of using Machine learning based IDS over conventional signature based IDS are as follows:

- It is easy to bypass the signature based IDS by doing slight variations in an attack pattern whereas Machine learning based IDS based on supervised techniques can easily detect the attack variants as they learn the behavior of the traffic flow.
- The CPU load is low to moderate in Machine learning based IDS as they do not analyze all signatures of the signature database as done by signature based IDS.
- Some of the Machine learning based IDS, particularly based on unsupervised learning algorithms, can detect novel attacks.
- Machine learning based IDS can capture the complex properties of the attack behavior and improve the detection accuracy and speed than conventional signature based IDS.
- Different types of attacks keep on evolving. Signature based IDS will require the maintenance of the signature database time to time and keep it up-to-date whereas Machine learning based IDS based on clustering and outlier detection won't require such update.

In this paper, we have mainly focused on the use of machine learning for anomaly, misuse or hybrid detection mechanism with their detailed analysis and investigated their capability for attack detection. A detailed study of various machine learning approaches is helpful in exploring solutions for advanced cyber intrusion detection. The machine learning based intrusion detection approaches have been categorized into four types. These are as follows: (i) Single classifiers with all features of data set (ii) Single classifiers with limited features of data set (iii) Multiple classifiers with all features of data set and (iv) Multiple classifiers with limited features of data set. In single classifier system, an individual classifier is used to detect Intrusions. Multiple classifier is a broad term which considers a set of ML algorithms at the time of learning and detecting intrusions. A set of classifiers are integrated to provide a common output for detecting intrusions. For example, Kim *et al.* [28] proposed multiple classifier methods which hierarchically integrates misuse detection model with anomaly detection model rather than just combining their results. DT C4.5 acts as a misuse detection module, and one class SVM acts as an anomaly detection module. Multiple classifiers based approach lower the false alarm and improve the detection rate. Each of these approaches learns from the available dataset which is described by a set of connection features such as source/destination port number, source/destination IP address, source bytes and destination bytes, etc. Ensemble learning is one of the forms of Multiple learning algorithms in which predictions by a set of classifiers are combined in some way, discussed in detail in Section IV.

Our analysis reveals that a feature set for analyzing the behavior of some particular category of attack, is different from the feature set of another category of attacks; since each of the attack categories posses some unique characteristics. We discuss the standard feature selection methods used by researchers if the domain knowledge of the attacks is not known. Our main goal of the paper is to perform a detailed investigation and critical analysis of using machine learning approaches for intrusion detection in environment. The performance analysis of various categories of machine learning techniques is also carried out, and observations are provided concerning each category. Our paper mainly concentrates on intrusion detection in wired cyber traditional networks.

The detailed discussion of intrusion detection application in wireless networks can be referred from here [29]–[31]. Different types of machine learning based IDSes are available for mobile devices. For example, *AmoxID* [32] is based on SVM algorithms and implemented for iOS and Android OS. *SMARTbot* [33] is an off-device behavioral analysis framework based on Artificial Neural Networks back-propagation method for mobile botnet detection and achieves 99.49% accuracy. A light-weight Android malware detection system is proposed by Shabtai *et al.* [34], called Andromaly which also uses machine learning algorithms. Sikder *et al.* [35] propose a context-aware sensor based attack detector, called *6thSense* to detect attacks which bypass the flaws in sensor management system. It makes use of Markov Chain, Naive Bayes, and Logistic Model Tree (LMT). A detailed survey of various types of machine learning based IDS for mobile devices particularly mobile phones can be found here [36]. A detailed survey on virtualization based attacks such as VM Escape, Side Channel Attacks, Hyperjacking, attacks on Guest-OS, etc. and their detection techniques in Cloud/Virtualization environment, has been separately addressed in our recent work [37]. More specifically, a survey on cache based side channel attacks and prevention approaches has also been recently published by Anwar *et al.* [38].

The major contributions of our present research work are as follows:

- The classification of attacks based on their characteristics is presented. Various factors that make the detection of low-frequency attacks (like U2R and R2L, Worms, ShellCode etc.) difficult to achieve by machine learning techniques are discussed and methods are suggested for improving their detection rate.
- The discussion of various existing literature for intrusion detection is provided, highlighting the key characteristics, the detection mechanism, feature selection employed, attacks detection capability.
- The critical performance analysis of various intrusion detection techniques is provided with respect to their

attack detection capability. The limitations and comparison with other approaches are also discussed. Various suggestions are provided for improvement in each category of techniques.
- Future directions of machine learning are provided for intrusion detection applications.

The paper is organized into XI Sections. In Section II, a comparison with related surveys is given, highlighting our specific contributions to compare our work. In Section III, a detailed description of different types of attacks with their characteristics is provided. In Section IV, the description of various machine learning techniques & their characteristics is presented with a discussion on the importance of feature selection in machine learning. Section V provides the detailed and comprehensive summary of different machine learning approaches for intrusion detection, and Section VI classifies them based on their ability to detect an attack. Section VII discusses the performance analysis of some machine learning techniques for detecting different security attacks. The security issues associated with each category of machine learning techniques are discussed, and solutions are provided to overcome the security issues. Various useful measures are provided for improving their detection rate followed by Section VIII, which describes the issues in detecting low-frequency attacks. In Section IX, various data mining tools for machine learning and deep learning have been discussed. In Section X, future directions are provided to give a brief insight into the ongoing and future research work. In the end, in Section XI, concluding remarks are mentioned with the scope of future work.

II. Related Work

There are surveys on applying machine learning to intrusion detection. Some of them are discussed to highlight their contributions. The specific contributions which make our work different than others are also presented. Agrawal and Agrawal [39] have provided a survey on anomaly detection using data mining techniques for intrusion detection. They have categorized the anomaly detection approaches based on three factors: clustering based approaches, classification based approaches and hybrid approaches. K-means. K-Meoids, EM clustering, Outliers detection algorithms have been described under clustering based approaches. Naive Bayes Algorithm, Genetic Algorithm, Neural Networks, Support Vector Machine have been described under classification based approaches. Hybrid approaches describe the combination of machine learning approaches. They have provided the brief comparison of papers using the ensemble based approaches.

Haq *et al.* [40] provided a survey on the application of machine learning techniques in intrusion detection. They have broadly classified the techniques into three major categories: supervised learning, unsupervised learning and reinforcement learning. In supervised learning, a classifier is trained on the labeled dataset. Unsupervised learning is used when we do not have the labeled dataset. In Reinforcement learning, a domain expert can label the unlabeled instances. They have provided a brief description of various single classifier and ensemble algorithms and provided references to papers using machine learning for intrusion detection without giving any critical analysis or observations.

Ahmed *et al.* [22] provided a survey on network anomaly detection approaches. Attacks are classified into four categories: DoS, probe, U2R and R2L based on KDD'99 dataset [12]. Each category points to a specific type of anomaly. They have provided the discussion over different types of machine learning approaches, i.e., classification based, clustering based, statistical based and information theory based approaches. The application of various types of machine learning approaches in intrusion detection, distinguishes the normal instances from anomalous instances. A brief summary of issues with various network intrusion detection dataset is discussed. The collaborative IDSes are suggested as a future research directions. However, a detailed in-depth description and analysis of various existing IDS proposals based on machine learning is lacking in their survey. Authors have also not provided future directions for machine learning algorithms.

A discussion on machine learning and data mining techniques for intrusion detection has been given by Buczak and Guven [41]. Their survey describes the application of machine learning and data mining techniques for misuse and anomaly detection. They have clarified the difference between machine learning (ML) and data mining (DM) and stated that ML is an older sibling of DM. Since they both use same methods for classification or knowledge discovery of data, they use the term ML/DM methods for algorithms under study. In their survey, they have described various methods and related them to misuse, anomaly and hybrid detection techniques. The description about the time complexity of algorithms is also mentioned in the paper. They have observed that KDD'99 and DARPA have been mostly used data sets as this makes the comparison relevant to authors. However, some researchers have used NetFlow and tcpdump dataset. They have recommended which ML/DM method will be suitable for misuse and anomaly detection individually.

In our survey, a detailed investigation and analysis of various machine learning techniques have been carried out for finding the limitations associated with various machine learning techniques in detecting intrusive activities. The key factor which differentiates the present work from existing surveys is that it is based on the premise that no one particular intrusion detection technique, based on single/multiple classifier algorithms can help in detecting all types of attacks. Hence, the use of specific intrusion detection technique is recommended for detecting a specific set of attacks. The importance of various factors while selecting an algorithm is discussed. Attack classification is also provided with attack examples and the specific attack features are mapped to each attack. Various issues in detecting low-frequency attacks are also mentioned and methods are also suggested for improvement.

A summary of various intrusion detection approaches is discussed including the literature on diverse datasets. In agreement with Buczak and Guven [41], we also found that people have mostly used KDD'99, DARPA dataset. Existing intrusion detection approaches based on machine learning techniques have been thoroughly analyzed concerning individual attack categories. Limitations associated with approaches

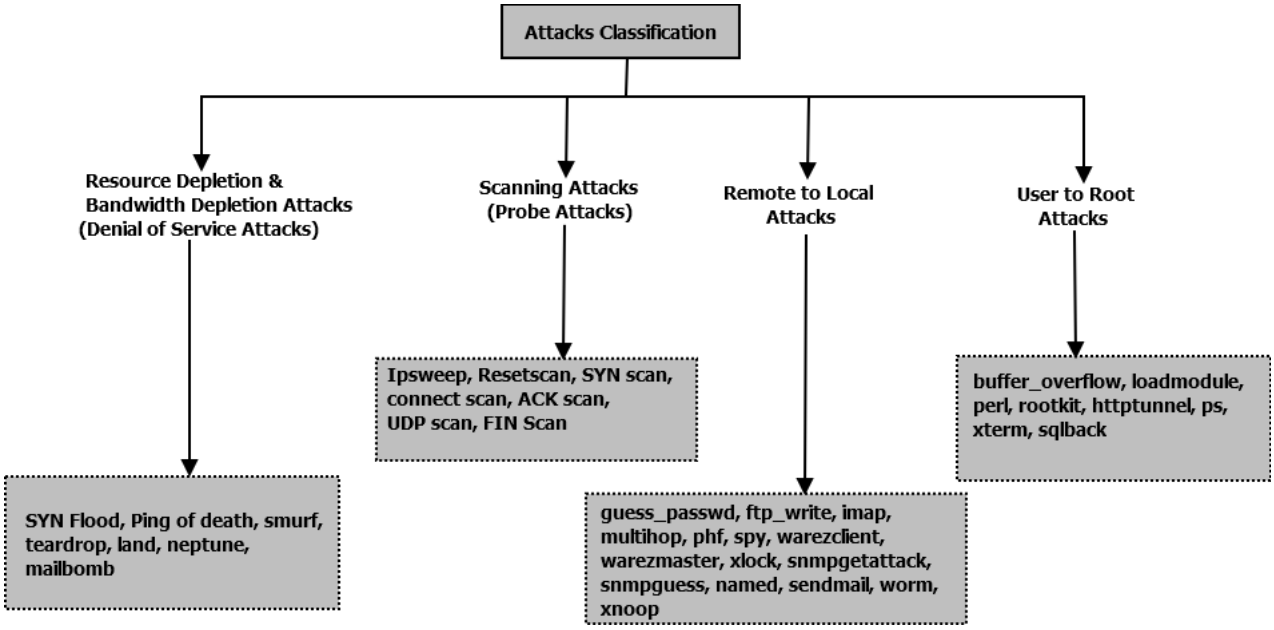

Fig. 1. A taxonomy of various Attacks based on KDD'99 dataset.

for each category are discussed with viable solutions. After the exhaustive survey of literature and critical analysis using the comparison of results reported by researchers, various observations are reported and analyzed. Future directions are provided in the field of intrusion detection. Future directions specifically point towards the usage of deep learning and reinforcement learning techniques for intrusion detection. Various challenges associated with these approaches have also been discussed in paper.

III. Classification of Attacks With Relevant Attack Features

Network and host based attacks have become pervasive in today's world. Attackers attempt to bypass the security of the network by exploiting the existing vulnerabilities in the network. They disturb the normal functioning of the network by malfunctioning the network devices, flooding the network by sending excessive packets, performing the scanning over a network, etc. It causes the unavailability of service to the legitimate users and highly reduces network throughput. Host based attacks attempt to bypass the security of a host machine. Attacker gains unprivileged access to a machine and tries to gain root access which may lead to the destruction of important system files, modification of sensitive data, leakage of user's private information, etc. Host based attacks can be launched as the next step after network attacks. Any machine over the network can be compromised by a hacker who has access to the network. In this case, the attacker first tries to establish the connection over the network to the target machine by exploiting the weakness in the network protocols or security devices such as Firewall [42], Intrusion Detection System [43] and then tries to copy malicious files over the network to the host machine. Once the user executes these files, the system is compromised and is under the control of the attacker. Now the attacker can perform any activity on these compromised hosts such as execute malicious programs and damage the system. Hackers exploit vulnerabilities present in the computer or network by using specialized tools such as Nmap, scapy, Metasploit, Armitage, Dsniff, Tcpdump, Net2pcap, Snoop, Ettercap, Nstreams, Argus, Karpski, Ethereal, Amap, Vmap, TTLscan and Paketto, etc. [44]–[47]. A detailed description of these tools can be found in [48]. In a secure environment, both network and host based security are important. In this Section, we have described attacks which are classified into three broad categories based on their characteristics as shown in Figure 1 and Figure 3. In each category, we have also described the important attack features for each attack based on KDD'99 dataset [12] and UNSW-NB dataset [13]. All these features are described in detail in Table II and Table III.

A. Denial of Service Attacks (Resource Depletion and Bandwidth Depletion Attacks

This category of attacks cause the unavailability of service to the legitimate users and hence also referred as DoS (Denial of Service) attacks [49]. For example, lets take an attack scenario: An attacker can send multiple service requests either to register with the enterprise or to access any of the valid service instance running in the enterprise. In this case, the administrative server will be flooded with many service requests and will fail to provide services to other legitimate customers/users. There can be another attack scenario where multiple machines are used to launch DoS attack: A large number of machines are connected to an organization or enterprise network. If an attacker has access to one or more machines of an organization/enterprise. It can misuse this privilege and can launch DoS attack to the other machines in the same network subnet. Here, the attacking surface is very broad, an attacker can occupy multiple machines (Zombies) and can

TABLE II
TCP CONNECTION FEATURES IN KDD'99 [50]

Category Name	Features Names	Type	Description
Basic Features	P1. Duration	Integer	Duration of the connection (seconds)
	P2. Protocol_type	Nominal	Type of Protocol (TCP, UDP, ICMP etc.)
	P3. Service	Nominal	Network Service (http, telnet, http, others)
	P4. Flag*	Nominal	Connection status (SF, S0, S1, S2, S3, OTH, REJ, RSTO, RSTO,S0, SH, RSTRH, SHR)
	P5. Src_bytes	Integer	Number of bytes sent from source to destination
	P6. Dst_bytes	Integer	Number of bytes received from destination
	P7. Land	Binary	If source and destination IP are identical. It is 1 else 0
	P8. Wrong_fragment	Integer	Sum of Bad checksum packets in a connection
	P9. Urgent	Integer	Some of packets where urgent bit is set 1.
Content Feature	P10. Hot	Integer	Sum of hot actions in a connection (entering a system directory, creating programs and executing programs)
	P11. Num Failed Login	Integer	Number of failed logins in a connection
	P12. Logged in	Binary	1 if successful login else 0.
	P13 Num compromised	Integer	Sum of not found error appearances in a connection
	P14. Root shell	Binary	1 if root shell is obtained else 0
	P15. Su attempted	Binary	Its 1 if su command attempted else 0
	P16. Num root	Integer	Sum of operations performs as a root in a connection
	P17. Num file creation	Integer	Sum of file creations in a connection
	P18. Num shells	Integer	Number of shell prompts
	P19.Num access files	Integer	Sum of operations in control files in a connection
	P20.Num outbound cmds	Integer	Sum of outbound commands in a ftp session
	P21.Is hot login	Binary	If the user is accessing s root , it is set to 1 else 0
	P22.Is guest login	Binary	If the user is accessing s guest, it is set to 1 else 0
Traffic Feature (2s time window)	P23. Count	Integer	Sum of connections to the same destination IP
	P24. Srv count	Integer	Sum of connections to the same destination Port no.
	P25. Serror rate	Real	Percentage of connections that have activated the flag (P4) s0, s1, s2 or s3, among the connections aggregated in count (P23)
	P26. Srv serror rate	Real	Percentage of connections that have activated the flag (P4) s0, s1, s2 or s3, among the connections aggregated in srv_count (P24)
	P27. Rerror rate	Real	Percentage of connections that have activated the flag (P4) REJ, among the connections aggregated in count (P23)
	P28. Srv rerror rate	Real	Percentage of connections that have activated the flag (P4) REJ, among the connections aggregated in count (P23)
	P29. Same srv rate	Real	Percentage of connections that were to the same service, among the connections aggregated in count (P23)
	P30. Diff srv rate	Real	Percentage of connections that were to different services, among the connections aggregated in count (P23)
	P31. Srv diff host rate	Real	Percentage of connections that were to different destination machines among the connections aggregated in srv_count (P24)
Traffic Feature (2s time window from dest. to host)	P32. Dst host count	Integer	Sum of connections to the same destination IP address
	P33. Dst host srv count	Integer	Sum of connections to the same destination port number
	P34. Dst host same srv rate	Real	Percentage of connections that were to the same service, among the connections aggregated in dst_host_count (P32)
	P35. Dst host diff srv rate	Real	Percentage of connections that were to different services, among the connections aggregated in dst_host_count (P32)
	P36. Dst host same src port rate	Real	Percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_count (P33)
	P37. Dst host srv diff host rate	Real	Percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count (P33)
	P38. Dst host serror rate	Real	Percentage of connections that have activated the flag (f4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (P32)
	P39. Dst host srv serror rate	Real	Percent of connections that have activated the flag (P4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_count (P33)
	P40. Dst host rerror rate	Real	Percentage of connections that have activated the flag (P4) REJ, among the connections aggregated in dst_host_count (P32)
	P41. Dst host srv rerror rate	Real	Percentage of connections that have activated the flag (P4) REJ, among the connections aggregated in dst_host_srv_count (P33)

*State Meaning [51]: SF Normal SYN/FIN completion, REJ Connection rejected, Initial SYN elicited a RST in reply
S0 State 0: initial SYN seen but no reply, S1 State 1: connection established (SYN's exchanged), nothing,further seen,
S2 State 2: connection established, initiator has closed their side, S3 State 3: connection established, responder has closed their side,
RSTO Connection reset by the originator, RSTR Connection reset by the responder, OTH Other, a state not contemplated here.

use them to launch DoS attacks. This kind of DoS is also called as Denial of Service attack (DDoS). DoS attack is classified into two types: Bandwidth Depletion and Resource Depletion attacks. In Bandwidth Depletion attack, attacker tries to overload the network by network packets. There are two classes in Bandwidth Depletion attacks: Flooding attacks and Amplification attacks. In Flooding attacks, attacker tries to flood the network by sending excessive ICMP or UDP packets causing overloading of the network resources. In Amplification attacks, attacker tries to exploit the IP address broadcast feature of most of the routers. This feature allows a sending system to specify a broadcast IP address as the destination address rather than a specific address. Example of such attacks are smurf and fraggle attacks [52]. In Resource Depletion attacks, attacker ties up the resources of a victim system. This attack can be launched by exploiting the network protocol (ex. neptune, mailbomb) or by forming malformed packets (ex. Land, Apche2, Back, teardrop, ping of death etc) which are sent to the victim machine over the network. A brief explanation about some of these attacks [53] is given below:

Land: In Land attack, an attacker sends spoofed SYN packet in which the source address is the same as the destination address. It is effective in some of the TCP/IP implementations.

TABLE III
TCP CONNECTION FEATURES IN UNSW-NB

Traffic Type	Feature Name	Type	Description
Flow Features	srcip	N	Source IP address
	sport	I	Source Port address
	dstip	N	Destination IP address
	dsport	I	Destination Port number
	proto	N	Transaction protocol
Basic Features	state	N	The state and its dependent protocol, e.g. ACC, CLO, else (-)
	dur	F	Record total duration
	sbytes	I	Source to destination bytes
	dbytes	I	Destination to source bytes
	sttl	I	Source to destination time to live
	dttl	I	Destination to source time to live
	sloss	I	Source packets retransmitted or dropped
	dloss	I	Destination packets retransmitted or dropped
	service	N	http, ftp, ssh, dns ..,else (-)
	sload	F	Source bits per second
	dload	F	Destination bits per second
	spkts	I	Source to destination packet count
	dpkts	I	Destination to source packet count
Content Features	swin	I	Source TCP window advertisement
	dwin	I	Destination TCP window advertisement
	stcpb	I	Source TCP sequence number
	dtcpb	I	Destination TCP sequence number
	smeansz	I	Mean of the flow packet size transmitted by the src
	dmeansz	I	Mean of the flow packet size transmitted by the dst
	trans_depth	I	the depth into the connection of http request/response transaction
	res_bdy_len	I	The content size of the data transferred from the servers http service.
Time Features	sjit	F	Source jitter (mSec)
	djit	F	Destination jitter (mSec)
	stime	T	record start time
	ltime	T	record last time
	sintpkt	F	Source inter-packet arrival time (mSec)
	dintpkt	F	Destination inter-packet arrival time (mSec)
	tcprtt	F	The sum of synack and ackdat of the TCP.
	synack	F	The time between the SYN and the SYN_ACK packets of the TCP.
	ackdat	F	The time between the SYN_ACK and the ACK packets of the TCP
Additional Features	is_sm_ips_ports	B	If source (1) equals to destination (3)IP addresses and port numbers (2)(4) are equal, this variable takes value 1 else 0
	ct_state_ttl	I	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
	ct_flw_http_mthd	I	No. of flows that has methods such as Get and Post in http service
	is_ftp_login	B	If the ftp session is accessed by user and password then 1 else 0.
	ct_ftp_cmd	I	No of flows that has a command in ftp session.
	ct_srv_src	I	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
	ct_srv_dst	I	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
	ct_dst_ltm	I	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
	ct_src_ ltm	I	No. of connections of the same source address (1) in 100 connections according to the last time (26).
	ct_src_dport_ltm	I	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
	ct_dst_sport_ltm	I	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
	ct_dst_src_ltm	I	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).

Attack Features: The attack can be detected by considering the feature 'Land'. If the value of feature 'Land' is 1, it means that source and destination address are identical. Hence this feature is most important in recognizing this attack.

Teardrop: In this attack, the attacker tries to send the fragmented packets to a target machine. He sets the fragment offset in such a way that the subsequent packets overlap with each other. If there is a bug in the IP fragmentation reassembly code of receiving target operating system, the machine crashes due to improper handling of the overlapping packets. Such attacks are successful on different operating systems such as Windows 3.1x, Windows 95, Windows NT and versions of the Linux kernel prior to 2.1.63.

Attack Features: Feature 'Wrong Fragment' which is the sum of bad checksum packets in a connection provides some clue about the malformed IP packets. Hence this feature is important in recognizing this attack.

Smurf: Smurf attack is an amplification based denial of service attack in which attacker sends a large number of ICMP echo messages to a broadcast IP address with the spoofed address of victim's machine as a source IP. On receiving the packet, each machine in the broadcast network replies to the victim's machine making its resources busy uselessly [54].

Attack Features: This attack can be easily detected in the victim machine by looking at the huge number of ICMP echo replies to victim machine without sending any ICMP echo requests packets from the victim machine. There are some feature such as 'Service' (ICMP), 'Duration', 'Dst host same srv rate' (used to find the percentage of connections to the same service and to the same destination IP address coming from attacker's machines) and 'Same srv rate' (used to find the percentage of connections to the same service and to the same destination IP address going from victim machine) which are useful in determining the total number of ICMP echo packet to victim machine within some duration of time and total ICMP reply packets from the victim machine within some duration of time.

Ping of Death: Ping of Death (PoD) is a denial of service (DoS) attack caused by an attacker deliberately sending an IP packet larger than the 65,536 bytes allowed by the IP protocol. The maximum allowable IP packet size is 65,535 bytes, including the packet header, which is typically 20 bytes long. This causes the system to crash or freeze. Many operating systems are vulnerable to this attack [55].

Attack Features: An attempted Ping of Death can be identified by noting the size of all ICMP packets and flagging those that are longer than 65,535 bytes. Features 'Dst bytes' (total number of bytes received) and 'Duration' in a connection may be helpful in providing some clue about PoD attack which means by comparing the total number of bytes received within a short duration of time with some threshold value (65,535).

Mailbomb: In Mailbomb attack, unauthorized users send a large number of email messages with large attachments to a particular mail server, filling up disk space resulting in denied email services to other users [56].

Attack Features: This attack can be identified by looking for thousands of mail messages coming from a particular user within a short period of time. Features such as 'Destination IP', 'Dst bytes' (total bytes received), 'Service' (SMTP/MIME), and 'Dst host same src port rate' (percentage of connections to the same port and to the same destination IP address) are important features in detecting the behavior of this attack.

SYN Flood: In SYN flood, TCP/IP implementation is exploited. An attacker sends the SYN request to the victim machine. Victim replies by ACK and waits for the reply. The server adds the information of each half-open connection in the pending connection queue. The half-open connections on the victim server system will eventually fill the queue and the system will be unable to accept any new incoming connections [57].

Attack Features: A SYN flood attack can be distinguished from normal network traffic by looking for a number of simultaneous SYN packets destined for a particular machine that are coming from an unreachable host or set a threshold for the duration of time a system has to wait for the reply. Hence features such as 'Duration', 'Flag' (S0: 'Initial SYN but no further communication' etc.), 'Dst host count' (percentage of connection to the same destination IP (victim machine)) are very important in recognizing this attack. Therefore noting those connections which are raising SYN flag with no connection established within a short duration of time are useful in detecting the attack.

B. Scanning Attacks

A scanning activity is a growing cyber security concern because it is the primary stage of an intrusion detection attempt that is used to locate the target systems in the network and subsequently exploit known vulnerabilities. An attacker sends a large number of scan packets to gain the detailed description about the machines using scanning tools such as nmap, satan, saint, msscan etc. Bou-Harb *et al.* [58] provided a detailed discussion on scanning techniques. They have provided a classification of cyber scanning topic into three parts: Nature, Strategy and Approach. The nature of scanning attack can be active or passive. The attack strategies could be remote to local, local to remote, local to local and remote to remote. They also classified 19 cyber scanning techniques with their pros and cons. At high-level all 19 categories are explained under five major categories: Open Scans [59], Half-Open Scans [60], Stealthy Scans [61], Sweep Scans [62] and Miscellaneous scans [63], [64]. For example, open scan and stealthy scan particularly SYN-ACK scan are shown in Figure 2 [58]. Open scan uses the TCP-handshake connection. It detects the TCP ports by making use of SYN flag and TCP protocols. A closed port replies with RST flag set (line i) whereas open port replies with ACK flag set (line ii). The attacker can now reset the connection by sending the RST and ACK. A firewall can detect such simple scans by looking at logs. Stealthy scan advances the open scans and also makes use of other flags together with SYN flag to avoid its detection. For the stealthy SYN-ACK scan, an attacker sends the SYN and ACK flag to the target, close ports sends the RST flag (line iii) whereas open ports will generate any response (line iv). It is a relatively fast method and does not require the three-way handshake or solo SYN flag. Other than scanning scenarios, the author also addresses the IP versions issues with cyber scanning activities. A separate literature review is provided for distributed detection techniques which are classified based on scanning activities one to one approach, one to many approach, many to one approach and many to many approach. These Probes are useful in launching future attacks [65]. Some of the scanning attacks are described below.

Ipsweep: An Ipsweep is used to determine which hosts are listening on the network by sending many ping packets. If a target host replies, the reply reveals the targets IP address to the attacker.

Attack Features: A Network Intrusion Detection System can examine the total number of ping packets coming within a short duration of time. Features such as 'Duration', 'Service' (ICMP), 'Dst host same srv rate' (used to find the connections to the same service) and 'Flag' (used to find connection status) are important to find the total ping messages within short

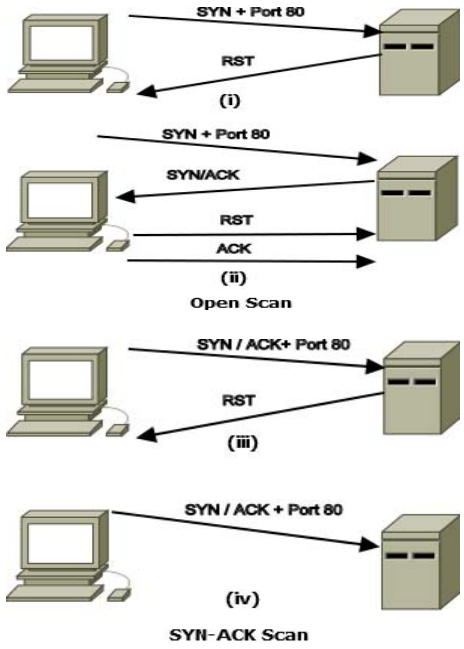

Fig. 2. Examples of scanning attacks: The Open Scan targeting (i) a closed and (ii) an open port (ii); The SYN|ACK Scan targeting (iii) a closed and (iv) an open port.

duration of time and current state of connection to detect the ipsweep attack.

Reset scan: In Reset Scan, an attacker sends reset packets (RST flag up) to victim machine to determine if the machine is active. If the victim machine does not send any response to reset packet; the machine is alive.

Attack Features: These scans can be detected by examining the various RST packets coming to a vulnerable machine with same service within a short period of time. Features such as 'Duration', 'Service', 'Flag', 'Dst host count' (used to find the sum of connections to vulnerable machine) are important to find the sum of connections that have initiated RST packet with short duration of time with same service protocol.

SYN scan: SYN scan is a half-open scanning attack because the attacker does not make a complete TCP connection. Attacker sends a large number of SYN packets to different ports. Open ports respond with SYN-ACK, and close port responds with RST.

Attack Features: These scans can be detected by checking the connections with large half-open connections with Flag either REJ (connections rejected; Initial SYN elicited, a RST reply) or S1 (SYN's exchanged nothing further seen) initiated by attacker machine. Hence, features such as 'Duration', 'Flag', 'Dst host diff srv rate' (percentage of connections to different ports and to the same destination IP) are important features to detect this attack.

C. User to Root Attacks

User to Root (U2R) attack refers to the group of exploits which are used to gain the root access to a machine by an unprivileged local user. These exploits are used in different ways to gain the root access to the machine. For example, in buffer overflow attack, the attacker exploits the vulnerability of a user program which copies too much data in a static buffer without checking to make sure the data will fit well. The attacker tries to manipulate the data that overflows the buffer and causes arbitrary commands to be executed by the operating system. In Ffbconfig attack [66], the attacker exploits the ffbconfig program distributed with some OS. Attacker overwrites the internal stack space of the ffbconfigct program which does not perform sufficient bound checking on arguments. The ffbconfig program is a part of FFB (Fast Frame Buffer) Graphics Accelerator. In loadmodule attack attempts to exploit the vulnerability present in some operating systems [67]. The loadmodule program loads to dynamically loadable kernel drivers into currently running system and creates to special devices in /dev directory. An attacker exploits the bug present in the loadmodule program to gain root access to the machine. Perl attack exploits the bugs saved in set-user-ID and set-group-ID scripts present in the Suidperl version of Perl. In this version, the interpreter does not exempt the root privileges properly when changing effective user and group IDs. Another example is rootkit attacks [68]. Rootkits are stealthy programs which are used to install a backdoor or hidden entry way to the attacker system to bypass the root privileges of the machine. Rootkits allow the attacker to hide many suspicious processes from the machine and install additional software such as sniffier, keylogger to compromise the resources of the machine [69].

Attack Features: KDD'99 features are not sufficient to observe the behavior of the attack. In fact, it is very difficult to distinguish these attacks from each other by considering the KDD'99 features. However, few features present in KDD 99 such as 'Num failed login', 'Su attempted', 'Is hot login', 'Num shells', 'Root Shell' and 'Num root', 'Duration' and 'Service' provides some hint about abnormal behavior of the root user and hence helpful in detecting U2R attacks.

D. Remote to User Attacks

Remote to User (R2L) attack refers to those group of exploits which are used to gain local access to the vulnerable machine, provided the attacker can send packets to the victim machine over a network. There are various ways to launch attacks to gain such an illegitimate privilege to a machine. In Dictionary/Guess Password attack, an attacker tries to make repeated guesses of possible username and password. The attack can be attempted using many services which provide the login facility such as telnet, ftp, pop, rlogin and imap. In FTPwrite attack, an attacker tries to exploit the ftp misconfiguration [70]. In ftp configurations, if ftp root directory or sub directories are not write protected and in the same group of the ftp account. An attacker can add files to these directories such as rhost files and gain access to the machine. In Imap attack, an attacker tries to exploit the buffer overflows of the Imap server which exists in the authentication code of the login transaction [71]. Attacker sends a carefully crafted text to execute arbitrary instructions. In Xlock attack, attackers exploit the unprotected X console of the user to gain access to the machine. Attacker displays the modified xlock program to the

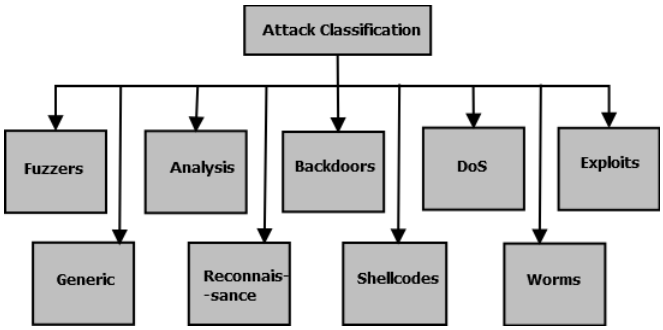

Fig. 3. A taxonomy of various Attacks based on UNSW-NB dataset.

user and waits till the user enters the password in that display. The password is sent back to the attacker by the trojan version of xlock program [72]. In wazermaster attack, an attacker tries to exploit the bug present in the FTP server. If FTP server has given write permissions to the guest account, an attacker can login to guest account in the public domain of FTP servers and can upload 'warez' (copies of illegal software) into the server. Users can later download these files [73]. Warezclient attack is launched by a legal user during FTP connection after the execution of warezmaster by an attacker. Users download the files (illegal software copies) from the server that were previously created by warezmaster [74].

Attack Features: Network connection features are not sufficient to observe the behavior of the R2L attacks. In fact it is very difficult to distinguish these attacks from each other by considering the network connection features. However, few features present in KDD'99 such as 'Duration', 'Service', 'Src bytes', 'Dst bytes', 'Num failed login', 'Is guest login', 'Num compromised', 'Num File creation', 'Count', 'Dst host count' and 'Dst host srv count' may provide some hint about abnormal behavior of a user in a local connection and hence helpful in detecting R2L attacks.

There is a little difference between R2L and U2R attacks. In U2R attacks, it is assumed that user has the local privilege to the victim machine (obtained via R2L attack). Attacker tries to attain the root privileges after accessing the machine. Hence the values of the traffic feature will be similar to the normal connection in case of U2R and least important to consider. The basic and content features are important in this case whereas in R2L attacks, attacker tries to obtain the local access to a remote machine. In R2L all features are important. In DoS and Probe attack, traffic features are very important together with other features. In Section VIII, we have described the difficulties in detecting these attacks using network attack data set.

On the basis of UNSW-NB attack dataset, attacks have been categorized into 9 types as shown in Figure 3. DoS is described earlier in detail. Other attacks are described below.

E. Fuzzers

In fuzzer attack, attacker sends a large amount of randomly generated input sequence from command line or in form of protocol packets. Attacker tries to discover security loopholes in the OS, program or network and make this resources suspended for a time period and can even crash them.

Attack Features: If a source is sending large number of packets continuously using same service protocol and/or at same destination port number over some duration of time. It could be indication of Fuzzer. In fact, some other features such as source to destination bytes, source of destination packet count and lots of variation (or 'jitter') can be the indication of problems. The features which are very helpful in this category of attack are: dur, service, sbytes, spkts, srcjitter, synack, cf_srv_src, ct_src_dport_ltm, described in Table III.

F. Analysis

This category of attack refers to various intrusions that penetrates the Web applications by various means such as port scanning, malicious Web scripting (like HTML files penetration) and sending spam emails etc.

Attack Features: The attack characteristics of various port scanning attacks and various important features for detecting those attacks are discussed in Section III-B. There are Anti-spam filters provided by the mail service providers to filter such emails coming from unauthorized source. Spam emails can be bypassed by such filters. Hence, in addition to source IP address, the analysis of overall network performance can be done by considering various possible features as listed in Table III.

In particular, Web application attacks can be detected by performing the HTML header, email header analysis or code analysis (scripting codes) [75].

G. Backdoor

In backdoor attack, attacker can bypass the normal authentication and can obtain unauthorized remote access to a system. Attacker tries to locate the data by doing fraudulent activities to bypass the system security of the system. Hacker uses backdoor programs to install the malicious files, modifying the code or gain access to the system or data.

Attack Features: Some of the important features that must be present in the feature set are as follows: {sport, dsport, dur, sbytes, service, ackdat, sjit, djit, ct_flw_http_mthd, is_ftp_login, ct_srv_src, ct_dst_ltm}. It won't be easy to get exact information about a backdoor attempt at victim machine. However, by analyzing network features, one can get some clue about unauthorized network attempts.

H. Exploits

Exploits category refers to intrusions that exploit the software vulnerabilities, bug or glitch within the operating system or software. Attackers utilize the knowledge of the software to launch exploits with an intention to cause harm to the system.

Attack Features: Various important features which are crucial for detecting the attempts of launching exploits at monitored machine are as follows: {srcip, dstip, sport, dsport, sinpkt, synack, is_sm_ips_ports, ct_ftp_cmd, res_bdy_len, ct_src_ltm, ct_src_ltm} (refer Table III). These features may provide some hint about the attempt of launching exploits. However, exploits can be more appropriately detected by monitoring the operating system behavior using dynamic analysis techniques. Once can refer our work for same [76], [77].

I. Generic

Generic attack against a cryptographical system, tries to break the key of the security system. It is independent of the implementation details of the cryptographic system. The structure of the block-cipher is not considered. For example, birthday attack is a Generic attack which considers hash function as a black box.

Attack Features: It would be good to take all possible network features of Generic attack into consideration. The accuracy of the system could not be very good if only considering network features. One can also perform the dynamic analysis of code to check the behavior of codes running in the victim machine. UNSW-NB does not provide system specific features such as root_login, su_attempted, Hot, Num_Shell etc. as specified by KDD'99.

J. Reconnaissance

Reconnaissance refers to attacks that gather information about the target computer network in order to bypass its security control. It can be defined as a probe which is a preliminary step towards launching further attacks. Attacker use port scanning OS scanning, nslookup, dig, whois, etc. to gather information about the system. Depending on TCP responses collected for each crafted packet we can make an intelligent guess of the operating system. After collecting sufficient information, attacks such as DDoS, worm, buffer-overflow exploits etc. can be launched.

Attack Features: Various important network features to detect such attacks: {sport, dsport, srcip, dstip, dur, spkts, sinpkt, service, synack, ct_srv_src, ct_src_ltm, ct_dst_ltm}. All the features provide the key network information about the source and destination system. The details about various port scanning attack, corresponding features and attack characteristics are already described earlier in Section III-B

K. Shellcode

A shellcode is used as a payload which is executed in the target machine to exploit the software's vulnerability. It is called as shellcode as it starts a command shell which is under the control of the attacker. Local shell codes try to exploit the vulnerability of high privileged process on a local machine for ex. bufferoverflow. Remote shellcode targets a vulnerable process running on a remote system. On successful execution, an attacker gains the remote access to the local machine. For ex. bindshell connects the attacker to a certain port of victim machine.

Attack Features: Some of the important features which are important for attack analysis are: {sport, dsport, srcip, dstip, dur, service, sbytes, dbytes, state, res_bdy_len, synack, is_ftp_login} (refer Table III). Network features may be helpful for detecting remote shellcode. However, in order to provide lower false alarms and good accuracy, shellcode can be detected by doing the behavior analysis of the programs. These types of attacks fall under the category of low-frequency attacks and can be launched easily at remote machines in a few attempts of making a network connection to the remote machine.

L. Worms

Worms are malicious programs or malware that replicate themselves and spread to other computers. It uses the network to spread the attack. Most of the worms are designed to replicate and do not try to change the system files. However, they can cause disruption to the services by increasing network traffic.

Attack Features: The important network feature could be as follows: srcip, dstip, sport, dsport, proto, spkts, dpkts, tcprtt, stcpb, dtcpb ct_srv_src, ct_flw_http_mthd, is_ftp_login etc, (refer Table III) which could help in analyzing the spread of packets from the same source address using particular service and Internet Protocol (IP) over a period of time.

Attacks are intentional attempts to destroy or gain unauthorized access to a machine or access user's data in an unauthorized way. Attacks target a computer network and/or a computer and harm the resources. Various attacks have been discussed in our study. Each attack is launched in some way and carries some unique characteristics which we have discussed. The network features which are essential for detection of a particular category of attacks have also been mapped to specific attack category. KDD'99 has been used by most of the researches. Hence, we have considered it for our attack study. However, as it is very old, we have also considered a very recent IDS attack dataset, i.e., UNSW-NB [13] which contains ten categories of attacks. ISCX-IDS attack dataset [78] is not publicly available. We obtained this dataset from University of New Brunswick (UNB) in the form of PCAP files on request. The attack features and their description is not provided by the authors. Hence, KDD'99 and UNSW-NB have been considered for the study.

IV. Machine Learning: Techniques and Feature Selection

In this Section, we have discussed various most popular machine learning techniques used for detecting Intrusions. These techniques hold different characteristics and provide different results for detecting intrusions. Here, we have mentioned the working of these techniques with their characteristics. We have further described the various features selection approaches with their pros and cons and provide the optimal feature set for each attack.

A. Techniques Used in Machine Learning

Machine learning techniques work in two phases: training and testing. In training phase, they perform the mathematical calculations over the training dataset and learn the behavior of traffic over a period. In the testing phase, a test instance is classified as normal or intrusive based on the learned behavior. Various popular machine techniques are described below.

1) Decision Tree: Decision tree learning methods use branching method to illustrate every possible outcome of a decision. They can work with discrete-value attributes and continuous value attributes as well. The learned trees are then represented in the form of if-then rules. Three basic elements of the tree are decision node, branch and leaf node as shown in Figure 4. Decision node specifies a test over some attribute.

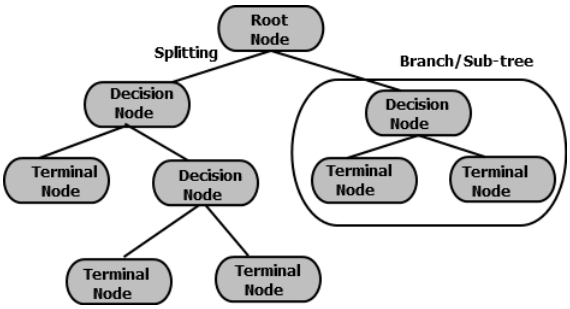

Fig. 4. Graphical representation of Decision Tree.

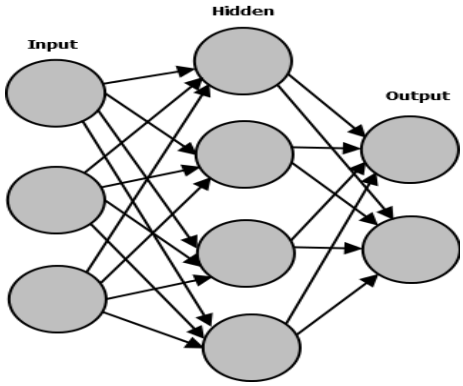

Fig. 5. Input, Hidden and Output layers in Neural Network.

Each branch represents one of the possible values for this attribute. At last, leaf node represents the class to which the object belongs. There exist various decision tree algorithms.

Some of the important decision tree algorithms are ID3 [79], C 4.5 [80], CART [81], LMT Tree, etc. ID3 is the very first DT algorithm developed by Quinlan. An ID3 algorithm uses greedy search approach. The tests are selected using information gain criteria. In ID3 algo, data may be overfitted and overclassified. ID3 does not handle missing values and numeric attributes. C4.5 is an improved version of ID3, given by Quinlan. It accepts both discrete and continuous values and splits the tree based on the gain ratio. It also solves the over-fitting problem by using error based pruning technique. J48 is an open source implementation of C 4.5 in Weka. It reduces the chances of overfitting. However, for noisy data, overfitting may happen. CART algorithm splits the tree based on towing criteria. It also handles both categorical and numerical values. It uses cost-complexity based pruning and handles missing values. Logistic Model Tree (LMT) uses a decision tree having linear regression model.

Most of these algorithms operate from root to leaf to arrive at some decision. The following measures are used for choosing the best attribute during classification: Entropy and Information gain. Entropy characterizes the impurity of an arbitrary collection of examples whereas Information gain measures how well a given attribute separates the training examples according to their target classification. A decision tree is suitable for the problems where (a) Instances can be represented by attribute-value pairs. Each attribute can have a disjoint set of possible values. (b) Target function should have discrete output value (for ex. yes or no). (c) The training data may have errors. Decision trees are robust to errors. (d) Training data may contain missing attribute values [82].

We have analyzed the performance of decision tree in Section VII. Decision trees perform better than other single classifiers as it implicitly performs the feature screening or feature selection based on the two parameters: Entropy and Information gain. The more Information gain a feature has, the more capable the feature is in discriminating the output classes. The top most nodes over which the tree split are the most important features of a dataset. They are not sensitive to outliers. However, computing probabilities of different possible branches, determining the best split of each node, and selecting optimal combining weights to prune algorithms contained in the decision tree are complicated tasks and involves high computational cost. Changing variables, excluding duplication information, or altering the sequence midway can lead to major changes.

2) Artificial Neural Network: Neural Network Learning methods provide a robust approach for approximating real-valued, discrete-valued and vector-valued target functions. The Multi layered Perceptrons Back Propagation Algorithm [83], Adaptive Resonance Theory based [84], Radial Basis Functions based [85], Hopfields Networks [86] and Neural Tree [87] are some examples of classification algorithms using Neural Network. ANN consists of three main elements: input node, hidden nodes (processing elements in hidden layers) and output node as shown in Figure 5. Multi-layer perceptron (MLP) neural network trained by Back-propagation learning (BPL) consists of two stages: feed forward and back propagation. Input data are fed to every node of hidden layer in feedforward stage. Each hidden node and output node calculates its activation value. The difference between the output target and the desired target value is used to generate an error. In back-propagation stage, the error is propagated back from the output layer to input layer, and weights are adjusted between output nodes and hidden nodes. The gradient descent method is used to update weights. The weights are updated till a predefined threshold is reached [88]. Neural Networks are suitable for the problems where a) Instances are represented by many attribute-value pairs. These values can be highly co-related or independent of each other. b) The target function output may be discrete-valued, real-valued or vector of real or discrete values. c) Training sample may contain errors. ANN is robust to noise. d) The learned function is typically difficult to understand by humans and this ability to understand the learned target function is not important by human [83].

Artificial Neural Network is a nonlinear model that is easy to use. BPL Neural Networks are easy to reach the local minimum and thus stability is lower. Especially for low-frequency attacks, the detection precision is low. It takes a longer time to train the neural network because of its nonlinear mapping of global approximation. Neural Network cannot detect temporarily dispersed and collaborative attacks because of inability to restore past events. It is difficult to find the accurate number of hidden layers and number of neurons. Classifier's performance also depends on the choice of the activation function. It

requires larger dataset and the output performance depends on the trained parameters and dataset relevant to the training.

3) Naive Bayes Classifier: Naive Bayes classifier is based on the Bayesian learning method and it is found to be useful in many applications. It is called "naive" because it is based on the simplifying assumption that attribute values are conditionally independent of each other. It is applied to the learning task where each instance x can be described by a conjunction of attributes and where the target function f(x) can take any of the value from some finite set V (a set of target values). In the learning steps various P(vj) and p(ai—vj) are estimated, given a training data $\{a_1, a_2, a_3 \ldots\ldots a_i\}$ of i attributes. It estimates the posterior probabilities of observing a class label from a set of normal class and anomaly class labels. For a given test instance, Class label with largest posterior is chosen as the predicted class [82]. It is suitable for the problems where a) Target function should have discrete output value (for ex. yes or no). b) Attribute-value pairs can represent instances. c) The independent assumption of Naive Bayes is acceptable. There are three Naive Bayes (NB) algorithms: Gaussian Naive Bayes [89], Bernoulli Naive Bayes [90] and Multinomial Naive Bayes [91]. Gaussian NB is used for continuous data values which are distributed according to a Gaussian distribution. Bernoulli NB is a binomial model used for binary feature vectors such as Bag of words model. Multinomial NB is used for discrete values in which feature vectors represent the frequencies in which certain events occur. The probability calculation is different in each of the three NB algorithms.

Naive Bayes classifier achieves a fast speed of detection and is simpler than other classifiers. However, it makes an assumption that features are independent of each other. This independent relation assumption may not hold true in detecting various types of attacks. For example, in the publicly available KDD'99 intrusion detection dataset, the features are highly dependent on each other. For example (refer Table II for features), feature P29 (same srv rate) is dependent on P23 (count). P23 refers to the sum of connections to the same destination IP address. P29 refers to the percentage of connections that were to the same service (tcp, http, icmp, etc.) among the connections aggregated in P23 (count). Similarly, feature P28 (srv error rate) is dependent on P23 (count). P27 (rerror rate) is also dependent on P23 etc. Such an assumption may not give desirable results for all types of attacks. Hidden Naive Bayes [92] is an extension of Naive Bayes and relaxes this assumption. It achieves an accuracy of (99.6%) for DoS attack detection.

4) Support Vector Machine: Support Vector Machine is one of the most successful machine learning technique in Intrusion detection when applied with other classifiers. SVM [93] is based on the notion of the margin-either side of the hyperplane that separates two data classes as shown in Figure 6. The generalization error can be reduced by maximizing the margin and creating the largest possible distance between the separating hyperplane and instances on either side of it. Data points that lie on the margin of optimum separating hyperplane are known as support vector points and solution is represented as a linear combination of these points. If the data contains misclassified instances, SVM may not be able to find the separating hyperplanes. One of the solutions to this problem is mapping

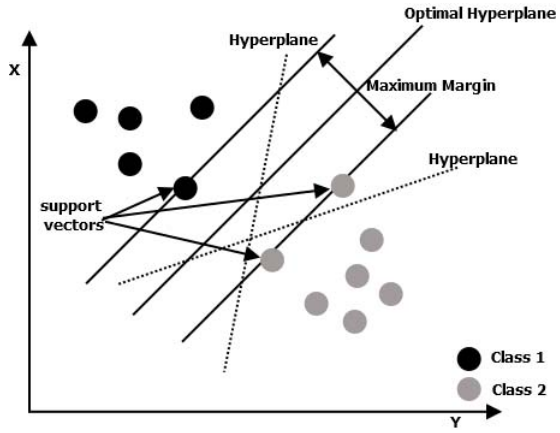

Fig. 6. Multi-class Linear SVM.

the data to a high dimensional space called 'feature space' and define separating hyperplane there. The kernel function is used to map the data into a new feature space for classification. The choice of a kernel function is very important here [94]. Radial basis kernel (RBF) [95] can be used to learn the complex region.

Therefore, SVM algorithms can be categorized into two types based on the type of kernel function: Linear SVM and Non-linear SVM. In Linear SVM, the training data is separated by hyperplane by the linear kernel function. If data is not linearly separable, nonlinear SVM classifier gives poor results [96]. Hence, nonlinear kernel maps the input data to a higher dimensional feature space to find the linear plane. Based on the type of detection (misuse/anomaly), SVM can be categorized into two types: Multi-class SVM and one-class SVM. Multi-class SVM is used for supervised learning algorithm. Multi-class classification using SVM can be done in two ways: one versus all (the traditional way) and one versus one. In one versus one, a set of binary SVM classifiers are built and the class is selected that is predicted by most of the classifiers. One class SVM is unsupervised machine learning algorithm used for novelty detection [97].

SVM suffers from the drawback of extensive memory requirement and algorithmic complexity. The performance also depends on the choice of the kernel function and choosing the parameters of kernel functions. Linear SVM produces less accurate results and produces overfitting. Training time of SVM is also very high which is not desirable in IDS where retraining a model is required time to time since user's behavior keeps on changing. Although it is robust to Noise.

5) Genetic Algorithms: Genetic Algorithms (GA) are search algorithms that find an approximate solution based on the principles of natural selection and genetics [98]. The four operators used in this process are initialization, selection, crossover, and mutation as shown in Figure 7. GA evolves to a high-quality population of individuals starting from the arbitrary selected initial population. Each is called a chromosome and is composed of a predefined number of genes. The quality of genes is measured by its fitness function and quantitative representation of each rule's adaptation [99]. During this process, the initially selected population is evolved for some

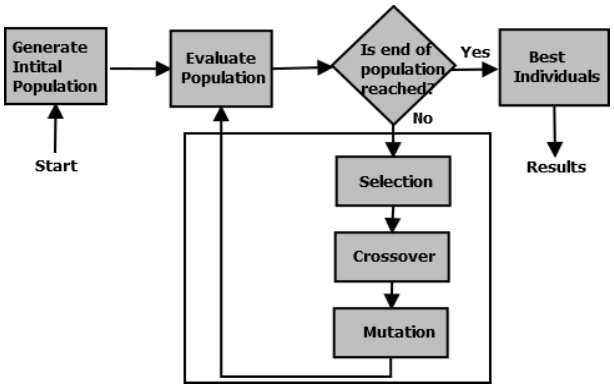

Fig. 7. Execution Flow of Genetic Algorithm.

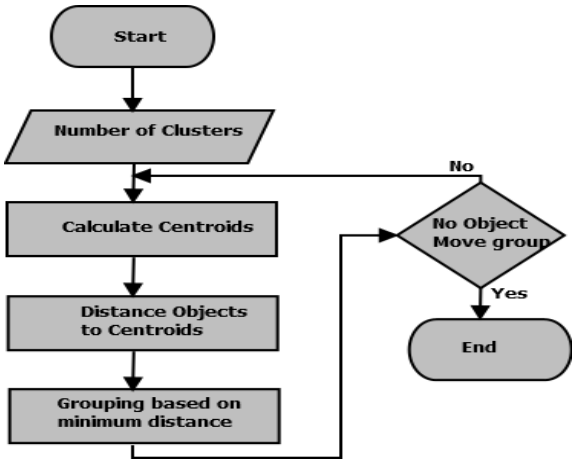

Fig. 8. Execution Flow of K-means Clustering.

generations. In each iteration, the three operators: selection, crossover, and mutation are sequentially applied to each with certain probabilities. Thus, only the fitness genes survive and reproduce. Some of the characteristics of GA are: a) They are intrinsically parallel, they can explore solution space in multiple directions at once. b) They are well suited for the problems where the space for the potential solutions is truly large. c) They are an adaptable system with GA and can be easily retrained which provides the possibility of generating new rules for Intrusion Detection. d) No gradient information is required. e) Do not require complex mathematics to execute. f) They work with a population of solutions rather than a single solution [100].

The three important things in Intrusion Detection are speed, accuracy and adaptability. GA has been observed to perform well when applied with other classifiers to optimize the parameters of classification process and in the selection of features in Intrusion Detection Systems. However, they lack in few aspects such as there is no absolute assurance that a genetic algorithm will find a global optimum. Moreover representing a problem space in the genetic algorithm is complex. They need a large number of fitness function evolution.

6) K-Means Clustering: K-means algorithm is a clustering based anomaly detection algorithms. They are based on the assumption that normal data instances lie close to their closest cluster centroid while anomalies lie far away from their closest cluster centroid [101]. In the first step, the data is clustered into K clusters assuming any K data points as the centroid of different clusters. The other data points are assigned to the clusters based on their closest distance measure from the centroids. In the next step, the centroid is recalculated as an average of data points of the cluster for each cluster. The process is repeated till some stopping criteria is reached such as till there is no change in centroid as shown in Figure 8. K-mean clustering has been widely adopted in integration with other classifiers by researchers working on Intrusion Detection Techniques. Some researchers have used it as a classifier to separate anomaly from normal data instance while some used it as a data compaction technique to separate outliers from the training data to provide the refined training data set to the classifier. In both the cases the detection results have been improved [102].

The techniques fail if the anomalies in the data form the clusters by themselves. In this case, this category of techniques will not be able to detect intrusions.

7) K-Nearest Neighbor Approach: K-NN is an instance based learning algorithms. KNN comes under nonparametric lazy learning algorithms [82]. Nonparametric means that it does not make any assumptions on the underlying data distribution. Lazy means it delays the generalization until the classification is performed. The training phase is much faster than other classifiers but more computational time is involved during the classification process. It is based on the assumption that instances in a dataset will exist in close proximity to other instances that have similar properties. It also assumes that normal data instances occur in dense neighborhoods while anomalies occur far from their closest neighbors. The label of the unclassified instance can be determined by looking at the class label of its neighbor instance. The Nearest Neighbour based anomaly detection techniques can be grouped into two broad categories: (1) Technique in which distance between a data point and kth neighbor is used as anomaly score. (2) Technique in which relative density of each data point is calculated as an anomaly score [103]. The choice of k affects the performance of kNN [104]. They are some important characteristics of kNN such as a) They are unsupervised and do not make any assumption regarding the generative distribution for the data. b) They are sensitive to the choice of similarity function which is used to compare instances. c) They require large storage. d) Computationally expensive technique. e) They are not robust to noise and can misclassify instances if noise is present [94].

Distance based kNN is used by most of the researchers in IDS to do the initial refinement of anomalies in the training dataset. However, the performance greatly depends on distance measure defined between a pair of data instances. It can be a challenging task to define the distance measure of the complex data. It fails to label the instances correctly if the normal data points do not have enough close neighbors while anomalies have enough close neighbors.

8) Fuzzy Logic: Fuzzy logic is a form of many-valued logic that deals with approximate rather than fixed and exact

reasoning. Fuzzy logic offers rigor of formal methods without requiring undue precision. It also offers alternative methods to handle policy preferences and conflicts [105]. A fuzzy set theory is defined in terms of fuzzy logic. Semantic of fuzzy operators are understood by using geometric model. Fuzzy logic can interpret the properties of a neural network and a precise description of its performance can be obtained. Neuro-fuzzy is very popular in the area of Intrusion Detection. It is applied by many researchers as discussed in Section V. A fuzzy set A in X is characterized by a membership function fA(x) which associates each point in X, a real number in the interval [0, 1], with the values of fA(x) at x representing the "grade of membership" of x in A. Thus, the nearer the value of fA(x) to unity, the higher the grade of membership of x in A [106].

Fuzzy logic is not enough to detect all types of attacks. It performs well when it is integrated with other classifiers. Fuzzy Logic techniques have been used in correlation with intrusion detection systems [107], [108]. The key characteristics of fuzzy logic are as follows [109]: (a) The fuzzy rules allow constructing the if-then rules which can be easily modified based on security applications. (b) They can combine the input from varying sources. (d) The quantitative measures used by IDS such as connection interval, CPU usage time, etc. are fuzzy in nature. (e) A numerical value can belong to multiple fuzzy sets at the same time, i.e., a numerical value does not have to be fuzzified using only one membership function. (f) The degree of alert that can be produced by an IDS is often fuzzy. The disadvantage of the fuzzy rules are as follows: (i) They consider that all factors are equally important which are to be combined. (ii) A fuzzy system requires more fine tuning and simulation before operational. (iii) It is hard to develop a model from a fuzzy system in comparison to other machine learning solutions due to the complexity involved in building the fuzzy model.

9) Hidden Markov Model: A Markov Model produces a behavioral model which is composed of states, transitions and actions. Both Hidden Markov Models (HMM) and Markov Chains come under the category of Markov Models. In Markov Chain, the transition probabilities are known which determine the topology of the model. In HMM, the system being modeled is represented by a Markov process with unknown parameters. HMM can be defined as a tool for presenting the probability distribution of a sequence. In HMM, an observation X_t at time t is generated by a stochastic process. However, state Z_t of the process cannot be directly observed (hidden). HMM satisfies the Markov property where the state Z_t depends only on the previous state Z_{t-1} observed at t-1 [110]. HMM maintains a transition matrix K*K. Each element A_{ij} of the matrix describes the probability of the transition from Zt-1 to Zt which can be written as: $A_{ij} = P(Z_{t,j} = 1|Z_{t,i} = 1)$. Ariu and Giacinto [111] proposed an HMM based IDS architecture known as HMMpayl which is an anomaly based IDS. The main goal of HMMpayl is to protect the Web server and the applications hosted by the server from attackers. A subset of n-grams are randomly selected from the sequence of payload and passed to HMM for further analysis. The advantage with the HMMpayl is that it reduces the computational cost in comparison to other n-gram models where all the possible sequences are analyzed. Authors have simulated the model using the HTTP traffic of DARPA'99 dataset. It trains k distinct HMM over the randomly generated subsamples of sequences. The output produced by each of HMM is finally combined to produce the detection accuracy. The ensemble of HMM found to perform better than single HMM classifier.

There are some advantages with HMM: (i) A HMM which is well tuned with parameters provide better compression than simple Markov Model. (ii) The model is fairly readable probabilistic graph model. (iii) HMM very well captures the dependencies between the consecutive sequences. (iv) The ensemble of HMM is found to perform well for recognizing the structure of sequences. There are some disadvantages with HMM: (i)'A fully connected HMM can lead to overfitting problem which happens when the model is trained with a dataset having large parameter space. (ii) HMM when implemented with the Viterbi algorithm, becomes expensive, both in terms of memory and compute time. However, Churbanov and Winters-Hilt [112] applied EM clustering with Viterbi to provide linear memory requirement.

10) Swarm Intelligence: A swarm can be considered as a group of cooperating agents which work together to achieve some purpose and task. Swarm Optimization is an advanced machine learning algorithm which is based on the evolutionary computations. Kolias *et al.* [113] provided a survey on swarm intelligence (SI) approaches for intrusion detection. They have provided a detailed comparison of various SI based IDS systems pointing to their advantages and disadvantages. The core SI based techniques used for supervised classification have been described. Most of the IDS described by authors are anomaly detection IDS. The SI based IDS approaches make use of multiple agents which collaborate with each other to solve a problem and provide the optimal solution. An agent can be used to find the classification rules for misuse detection or finding the clusters for anomaly detection. They have mainly categorized the SI based approaches into three types: (i) Ant Colony Optimization (ACO) based IDS, (ii) Particle Swarm Optimization (PSM) based IDS (iii) Ant Colony Clustering (ACC) based IDS. ACO algorithms are motivated by the behavior of the ants to find the shortest paths from their nest to the food. AntNag [114] is the first ACO algorithm for intrusion detection which is based on making directed graphs for attacks. PSO algorithms are motivated by the coordinated movement dynamics of animal groups. ACC algorithms are motivated by the clustering and sorting behavior of ants to work autonomously. A detailed study can be referred from their literature. There are some advantages with swarm optimization: (i) SI based systems are adaptable and can be adjusted to new stimuli. (ii) These systems are scalable since same control architecture can be applied to a group of agents. (iii) They are flexible as agents can be easily removed or added without affecting the architecture, etc. There are also some disadvantages for SI based systems: (i) The complexity associated with swarms provides the unpredictable results. (ii) A rich hierarchical swarm based system takes time to shift to states. (iii) There is no central control which makes the system redundant and uncontrollable, etc. [115]

11) Ensemble of Classifiers/ Ensemble Learning: Ensemble learning makes use of multiple learners and combines the predictions made by a set of classifiers called as base learners. The use of multiple machine learning algorithms helps in generating a set of hypotheses for a problem. The ensemble of classifiers integrates the hypotheses to generate a common result. The ensemble of classifiers provides a stronger generalization capability compared to individual base learners [116]. Each base learner is generated by using machine learning algorithms such as Decision Tree, Naive Bayes, Neural Network, Support Vector Machine etc. Some of the ensemble methods make use of the homogeneous base learners in which multiple instances of the same machine learning algorithm are used to generate a set of hypotheses over different sub-samples of the same training dataset. For example, Random Forest is one of the popular ensemble classifiers which combines the predictions made by the multiple decision trees. Some of the ensemble methods make use of the heterogeneous base learners in which different machine learning algorithms are used as base learners to generate a set of hypotheses. For example, Neural Network, SVM and DT can be trained over a training dataset and their predictions can be combined to generate common predictions. There are many ways to combine the predictions made by multiple base learners. Some of the popular methods are bagging, boosting, majority voting and stacking [117].

The first effective method of ensemble learning was Bagging, also called bootstrap aggregation. Bagging is used to reduce the variance of the machine learning algorithms having high variance. A variance can be termed as the amount of the change in the prediction of the target function for the different training dataset. Ideally, variance should not change too much. Bagging creates different instances of the same training dataset by selecting bootstrap samples. A bootstrap sample is created by selecting a sub set of samples from the training dataset. If a training dataset of size n is given; Bagging will generate m new training sets of size k by sampling with replacement. In sampling with replacement, an observation may be repeated multiple times in a set. Each of the newly generated training set is used to train a model. The output /predictions by each model can be combined by either voting (in case of classification) or averaging (regression).

In boosting, random samples of training dataset without replacement (any observation cannot be repeated in sub-set) is extracted and used to train a weak learner. In the second iteration, another sample sub-set is extracted in the same way; however, in the new training set 50% samples which are previously misclassified are added and another weak learner is trained. In next iteration, a new training set is formed having the samples which are classified by both the previous learners. After having some such iterations, the predictions made by weak learners are combined by the majority voting scheme. Both Boosting and Bagging combines the predictions based on the majority vote or average rule [118]. Stacking makes use of the combiner algorithm to combine the predictions. Stacking is a meta-algorithm which trains n base classifiers/learners for the given dataset. Each of the trained classifiers generates the predictions for the given input data. The output of each of the classifier is again learned by a combiner or meta-algorithm to make the final prediction. Ensemble classifier has some advantages over the single classifiers: (i) Generalization ability of the ensemble is much better than single classifiers.(ii) It may not be possible to select a particular learner based on the available training dataset. The search process may take longer. (iii) Ensemble often reduces the overfitting problem of single classifiers and improves the prediction error, etc. There are some disadvantages of ensemble learning is as follows: (i) The complexity of the ensemble affects the training time. (ii) Sometimes learning concepts become difficult to understand. (iii) Requires more memory than single classifiers, etc.

In this subsection, we discussed various machine learning techniques. Some of them are supervised such as various DT algorithms, multi-class SVM, MLP BP-ANN, NB and KNN, etc. Supervised machine learning algorithms can detect known attack patterns. They require a labeled attack dataset. Whereas some of the ML algorithms such as one-class SVM, K-Mean Clustering, Self Organizing Map (SOM), DBSCAN, etc., are some of the examples of unsupervised machine learning algorithms. Unsupervised learning is helpful in analyzing the unlabeled attack dataset and finding the outliers. The outliers can be noise or it can be anomalies which are rarely found in the normal scenarios. The outliers are further explored statistically and useful information is extracted out of them which can be helpful to find distinct characteristics from data. Fuzzy logic can be applied in Classification and Clustering algorithms to improve their learning capability and attack detection rate. For ex., Neuro-Fuzzy and Fuzzy c-means clustering are some popular applications of fuzzy logic in different ML techniques.

Swarm intelligence techniques such as Particle Swarm Optimization (PSO) is helpful in nonlinear optimization problems. HMM can be used to capture the dependencies between sequences using probabilistic approach. Both swarm intelligence and HMM can be used for supervised and unsupervised learning problems. Different classification algorithms have got different characteristics. Each one of them has some pros and cons as discussed before. The ensemble learning provides the combination of same/different supervised and unsupervised algorithms used to solve the target problem altogether. It often reduces the over-fitting problem of single classifiers and improves the classification rate.

B. Feature Selection in Machine Learning

Two main important things that highly affect the performance of a classifier are: Classifier's Technique and Selected Feature Subset. Researchers have proposed various combinations of classifiers and feature selection methods (discussed in detail in Section IV). The goal of feature selection is to select the most important and optimal subset of features. Features selection improves the generalization performance, reduces the computational cost of the classifier and makes the classifier faster for detecting unseen data and simplifies the understanding of data processing. There are various drawbacks of considering all features in the detection technique

such as (i) It will increase the computational overhead of the system and will make training and testing time slower. (ii) It will also lead to more storage requirement as the more number of features a database contains, the more space it requires to store each feature. (iii) It limits the generalization capability of a classifier which uses data mining techniques for detecting intrusions. (iv) It increases the error rate of the classifier since irrelevant features diminishes the discriminating power of relevant features.

The feature selection methods are categorized into three types [50]: (i) Filter methods (ii) Wrapper methods (iii) Hybrid/Embedded methods. Filter methods are independent of the classifier. They compute the intrinsic properties of the data. Filters are fast enough compared to the other methods. They are relatively robust against overfitting. The major drawback is that they do not consider the results of the classifier's performance over the selected features. Hence, they fail to provide the best feature subset for classification. Wrapper based methods use the combination of feature subset searching algorithms and classifier algorithm. The performance is measured as per the classification rate. The feature subset with good classification rate is chosen at the end. A classification rate threshold is taken into consideration as a stopping criteria of feature selection. Thus wrapper based methods are classifier dependent. The major drawback of this method is that successive learning of classifier may result in overfitting problem. Computational time is usually high as it involves successive iterations of subset selection algorithm and classifier. Hybrid methods are combined with the classifier's design in the training phase. Data exploitation is optimized which will reduce the number of retraining of the classifier for each new subset. Hybrid methods have higher computational cost than filter based methods. The importance of various feature selection algorithms for intrusion detection application has been addressed in detail here [119].

Let us take an example of features of KDD'99 network attack dataset. There are 41 features. Table II (shown in Section III) categories all features into four main categories:

- Basic Features (1-9): It refers to the basic features of an individual TCP connection such as P3 ('Service'), P1('Duration'), P4('Flag'), etc.
- Content Features (10-22): Content Features are extracted from the data portions of the packets such as P11 (num of failed logins), P14 (Root Shell), P10 ('Hot') and P13 ('Num Compromised'). These features are important to detect low-frequency attacks such as U2R and R2L. This is because DoS and Probe normally involves a large number of connections within a shorter period whereas R2L and U2R normally involves a single connection and are embedded in the data portion of a packet.
- Traffic Features (23-31): Traffic features are computed using a 2s time window such as P23 ('Count'), P24 ('Srv count'), P29 ('Same srv rate') etc. These are very important for detecting high-frequency attacks (DoS and Probe).
- Traffic Features (32-41): Traffic features are computed using a 2s time window from destination to host such as P32 ('Dst host count'), P33 ('Dst host srv count'), P34 ('Dst host same srv rate') and P39 ('Dst host srv serror rate') etc. These are very important for detecting high-frequency attacks (DoS, Probe).

We have provided an overview of different types of machine learning algorithms, i.e., rule based, probability based, clustering based, ensemble learning, genetics based, swarm-intelligence based, etc. The key characteristics, advantages and disadvantages of various machine learning algorithms have also been discussed. It would be impropriated to say that one type of machine learning algorithm will work best in all type of dataset. A classifier's accuracy is not solely important factor. Many factors affect the selection of appropriate classification model such as Is our data composed of categorical only, or numeric only, or both? What is the size of dataset?, Do we need to retrain the classifier often? Do we need quick and fast deployment model? Do we have labeled data or unlabeled data? What is the complexity of data?. Hence, the key characteristics of machine learning algorithms must be known before selecting a bunch of classifiers for performance analysis.

Also, the importance of feature selection and types of feature selection methods are also presented that can be used with machine learning algorithms. Feature selection helps in identifying important, non-redundant and relevant attributes that contribute to the accuracy of the predictive model. Also, considering less and important features speed up the classifiers.

V. Summary of IDS Based on Single/Multiple Classifier Based Machine Learning Approaches

Machine learning techniques have been used in different ways for detecting intrusions using publicly available datasets such as KDD'99 [12], DARPA 1998 [53]. These datasets have 41 features as shown in Table II in Section III. Recent attack datasets such as ISCX 2012 [78] and UNSW-NB15 [13] has been used in some of the approaches for validation. The summary of various IDS proposals is shown in Table IV and their performance results are shown in Table V. The acronyms for evaluation metrics are shown in Table VI.

Initially, single classifier techniques have been used as a standalone entity to classify the intrusions but they lack in performance for correctly classifying intrusions from normal data instances. Later, feature selection methods are proposed to improve the detection rate and to reduce the computational cost. However, there was no significant improvement in classification rate. Afterwards, some researchers combined the single classifiers to improve the detection rate of intrusions by using the 41 features of the dataset. There are some limitations with this technique such as low detection detect rate and high computational cost especially for the low frequency (which are in less frequency in the dataset) attacks such as low-frequency attacks. In some approaches, feature selection has been used with multiple Classifiers which greatly improved the detection rate for all types of attacks. However, there was no significant improvement in computational cost due to multiple processing of classifier modules. By the gradual evolution of machine

TABLE IV
SUMMARY OF EXISTING IDS APPROACHES BASED ON MACHINE LEARNING

Reference	ML Approach	Dataset	Features	Feature Selection approach	Attacks Detected	Performance	Problem Domain
Kim et al. [120]	SVM	KDD'99	-	–	DoS, Probe U2R, R2L	DR	Misuse Detection
Amor et al. [121]	NB vs DT	KDD'99	-	-	DoS, Probe U2R, R2L	DR	Misuse Detection
Zhang et al. [123]	Serial and multiple RBF NN & clustering	KDD'99	–	–	Smurf, Ipsweep, Guess Password Buffer Overflow	DR, FP	Misuse & anomaly Detection
Mukkamala et al.[124]	Ensemble of ANNs, SVMs and MARS	DARPA98	All	–	DoS, Probe, U2R, R2L	Accuracy	Misuse Detection
Bouzida et al. [122]	BPL NN, Enhanced DT C4.5	KDD'99	All	-	DoS, Probe, U2R, R2L	DR	Misuse Detection
Wang et al. [125]	PCA	DARPA98	All	Feature extraction (PCA)	DoS, Probe, U2R, R2L	DR, FPR, ROC	Misuse Detection
Khan et al. [126]	CT SVM (SVM with hierarchical clustering (DGSOT))	DARPA 1998	All	–	DoS, Probe, U2R,R2L	DR, Acc. FP,FN, TT	Hybrid Detection
Li et al. [127]	Active learning based TCM-KNN	KDD Cup 99	8, All	Feature Selection (Chi square)	DoS, Probe, U2R,R2L	TP, FP	Misuse Detection
Chen et al. [128]	FNT with PSO and GA	DARPA	DoS (12) Probe(12) U2R(8) R2L(10)	GA	DoS, Probe, U2R,R2L	DR, FP	Misuse Detection
Xiang et al. [129]	Bayesian clustering, DT C4.5	KDD'99	All, 4,14	Arbitrarily	DoS, Probe, U2R,R2L	DR	Hybrid Detection
Tong et al. [130]	RFB NN, Elman Network	DARPA	All	–	DoS, Probe,	DR, FPR	Misuse Detection
Tajbakhsh et al. [131]	Fuzzy Association rules	KDD'99	NA	Item reduction	DoS, Probe, U2R,R2L	DR, FP, Total time (TT + TeT)	Misuse Detection
Wang et al. [132]	FC-ANN	KDD'99	All	–	DoS, Probe, U2R,R2L	Precision, Recall, F-value. TT	Hybrid Detection
Sangkatsanee et al. [61]	Decision Tree C4.5	RLD09 (self) KDD'99	12	Feature selection (Information gain)	DoS, Probe	TDR, NDR, ATD, DDR, PDR	Misuse Detection
Amiri et al. [133]	MMIFS with LSSVM	KDD'99	8(DoS) 12(Probe) 14(R2L) 12(U2R)	Feature Selection (MMIFS)	DoS, Probe, U2R,R2L	Accuracy, DR, FPR, Building time, Testing TIme	Misuse Detection
Horng et al. [134]	Hierarchical Clustering and SVM	KDD'99	DoS(19) Probe(17) U2R(24) R2L(24)	Gradual Feature Removal	DoS,Probe, U2R,R2L	DR, Acc., FP	Hybrid Detection
Boughaci et al. [135]	FPSO	KDD'99	16	Manual selection	DoS,Probe, U2R,R2L	DR, TP, FN	Anomaly Detection
Lin et al. [136]	SVM,DT,SA	KDD'99	23	SVM, SA (simulated annealing)	DoS, Probe, U2R, R2L	Accuracy, Classification rate	Misuse Detection
Casas et al. [26]	Sub Space Clustering, DBSCAN , EAR	Public MAWI, KDD'99	9	Standard traffic features (known attacks)	DoS, Probe, U2R, R2L	ROC, Accuracy, Computation Time	Anomaly Detection
Sindhu et al. [137]	Wrapper (GA, Neuro tree and DT C4.5)	KDD'99	16 (not mentioned)	GA with Neuro Tree	Total 23 attacks (DoS, Probe, U2R,R2L)	TP, FP, Precision, Recall, F_measure	Misuse Detection
Li et al. [138]	K-mean ,ACO, SVM	KDD'99	19	GFR	DoS, Probe, U2R,R2L	DR, TT, TeT, Accuracy	Hybrid Detection
Chandrasekhar et al. [139]	K-means, Nero Fuzzy and SVM classifier	KDD'99	6	Neuro fuzzy	DoS, Probe, U2R,R2L	Accuracy, TP, FP, TN,FN	Hybrid Detection
kumar et al. [140]	Neural Network	KDD'99	All	–	DoS, Probe, U2R,R2L	DT, Acc	Misuse Detection
Kim et al. [28]	DT C4.5 and one class SVM	NSL KDD'99	All	–	DoS, Probe, U2R,R2L	ROC, TT, TeT	Hybrid Detection
Kuang et al.[141]	KPCA and SVM with GA	KDD'99	NA	Feature Extraction (KPCA)	DoS, Probe, U2R,R2L	DR, FAR, cc, TrD, TeD	Misuse Detection
Feng et al. [142]	CSVAC (SVM+CSOACN)	KDD'99	All	Data selection	DoS, Probe, U2R,R2L	DR,TT,FP,FN	Hybrid Detection
Koc et al. [143]	HNB	KDD'99	7	Feature Selection (INTERACT) Discretization method (PKID)	DoS, Probe, U2R,R2L	Accuracy, Error rate, cost	Misuse Detection
Lin et al. [144]	CANN with k-NN	KDD-Cup 99	6, 19	FEx(CANN)/Feature Selection (PCA)	DoS, Probe, U2R, R2L	Accuracy, DR, False alarm	Anomaly detection
Amoli et al. [145]	DBSCAN	ISCX	All	-	DoS, DDoS Probe	Accuracy, Rekall, Precision, TNR, FPR	Misuse Detection
Kumar et al. [146]	MOGA	ISCX	All	-	DoS, DDoS brute force SSH	Detection rate FPR	Misuse Detection
Toosi et al. [147]	Neural Fuzzy & GA	KDD'99	All	-	DoS, Probe, U2R or R2L	Detection rate FPR	Misuse Detection
Elhag et al. [148]	GFS, fuzzy association rules (FARC-HD)	KDD'99	All	-	DoS, Probe, U2R or R2L	Accuracy, detection rate false alarms	Misuse Detection
Yassin et al. [149]	K-means & NB	ISCX	All	-	DoS, DDoS, brute force SSH	Accuracy, detection rate false alarms	Hybrid Detection
Gupta et al. [150]	CRF model	KDD'99	different for each class	Attack knowledge	DoS, Probe U2R, R2L	detection rate false alarms	Misuse Detection
Mamun et al. [151]	LSSVM	ISCX	10	GA+LSSVM (Feat. selec.) shannon's entropy (Feat. Extrac.)	DoS, DDoS Brute force SSH	detection rate false alarms	Misuse Detection
Bhamare et al. [152]	Logistic Regression	UNSW-NB15	All	NA	DoS, Fuzzer, Analysis, etc.	Accuracy, FPR	Misuse Detection
Gharaee et al. [153]	LSSVM	UNSW-NB15	6-14	GA and SVM	DoS, Fuzzer, Analysis, etc.	Accuracy, TPR FPR	Misuse Detection
Chowdhury et al. [154]	SVM	UNSW-NB15	3	Simulated Annealing	DoS, Fuzzer, Analysis, etc.	Accuracy, FPR, FNR	Misuse Detection
Moustafa et al. [155]	EM, NB, LR	UNSW-NB15	-	CP, Apriori, ARM	DoS, Fuzzer, Analysis, etc.	Accuracy, FAR	Misuse Detection

learning techniques in intrusion detection, we have classified them into four categories. (i) Single classifiers with all features. (ii) Single classifiers with limited features. (iii) Multiple classifiers with all features (iv) Multiple classifiers with limited features. These techniques have been described in detail below.

TABLE V
PERFORMANCE RESULTS OF EXISTING IDS APPROACHES

Technique	Attack	Detection Rate (Attack Vise)	False Alarm	Accuracy	TPR	FPR	Overall Performance
SVM [120]	DoS, Probe, U2R, R2L	91.6, 36.65, 12, 22	–	–	–	–	–
NB and DT [121]	DoS, Probe, U2R, R2L	96.65 (NB), 97.24(DT) 88.33 (NB), 77.92(DT) 11.84 (NB),13.60(DT) 8.66(NB) 0.52(DT)	–	–	–	–	–
Serial and multiple RBF NN [123]	Smurf, Ipsweep, GuessPasswd, BufferOverflow	99, 99.5, 99.7, 98.8	0, 0.8, 4 3.3	–	–	–	–
Ensemble of ANN, SVM and MARS [124]	DoS, Probe, U2R, R2L	–	–	99.97, 99.85 76, 100	–	–	Acc=99.82
BPL NN, DT C4.5 [122]	DoS, Probe, U2R, R2L	97.00 (NN), 99.99 (DT) 71.63 (NN), 99.78 (DT) 0.0 (NN), 90.39 (DT) 26.68 (NN) 98.93 (DT)	–	–	–	–	–
PCA [125]	DoS, Probe, U2R, R2L	99.2, 80.7 88.5, 94.5	–	–	–	0.2 4 0.6 4	–
Neuro Fuzzy [147]	DoS, Probe, U2R, R2L	99.5, 84.1, 14.1, 31.5	–	–	–	–	DT=95.3 FA=1.9 CPE=0.1579
CT SVM [126]	DoS, Probe, U2R, R2L	97.35, 91.2, 17.23, 43.98	–	–	–	–	Accuracy=69.8% TT=13.18h Avg(FP)=37.8% Avg(FN)=29.8%
TCM-KNN [127]	DoS, Probe, U2R, R2L	–	–	–	–	–	TP=99.6%, FP=0.1%
FNT with GA and PSO [128]	DoS, Probe, U2R, R2L	98.75, 98.39, 99.70, 99.09	–	–	–	–	–
Bayesian clustering, DT C 4.5 [129]	DoS, Probe, U2R, R2L	98.66, 93.40, 71.43, 46.96	–	–	–	–	TT=86625s(24h) TeT=257s(4.28m)
RBF/Elman NN [130]	DoS, Probe	87.5, 100	0, 0	–	–	–	–
Fuzzy association [131]	DoS, Probe, U2R, R2L	78.9, 88.5, 68.6, 6.2	–	–	–	–	DR=70-90% FP=2%
FC-ANN [132]	DoS, Probe, U2R, R2L	99.91, 48.12, 93.18, 83.33	–	–	–	–	Precision=91.32% Recall=99.08% F-value=95.04%
Decision Tree C 4.5 [61]	DoS, Probe	99.434, 98.868	0.73, 0.9	–	–	–	TDR=99.4%, NDR=99.7%, ADR=99.0%, TT=0.28s
Hierarchical Clustering and SVM [134]	DoS, Probe, U2R, R2L	99.5, 97.5 19.5, 28.8,	–	–	–	–	Accuracy=95.7% FP=0.7%
FPSO [135]	DoS, Probe, U2R, R2L	97.22, 77.77, 69.44, 97.22	–	–	2.77, 0.0, 0.0, 2.77	–	–
Ensemble of SVM,DT,SA [136]	DoS, Probe, U2R, R2L	100, 98.35, 80, 90.67	–	–	–	–	Accuracy=99.96%
Sub Space Clustering, DBSCAN , EAR[26]	DoS, Probe, U2R, R2L	–	–	(KDD) 98, 100, 88, 85	–	–	TPR=0.94(MAWI) TPR=.90 (KDD) FPR=2 (MAWI) FPR=.0375 (KDD)
Wrapper (GA, Neuro tree and DT C4.5)[137]	DoS, Probe, U2R, R2L	–	–	–	–	–	DR=98.38% EP=1.62%
Kmean ACO SVM [138]	DoS, Probe, U2R, R2L	97.67, 91.45, 53.84, 90.34	–	–	–	–	TT=0.1183s TeT=4.63227s Accuracy=98.62%
K-mean, Neuro Fuzzy, Radial SVM[156]	DoS, Probe, U2R, R2L	97.64, 86.7, 3.83, 7.661	–	98.80, 97.31, 97.52, 97.51	–	–	–
Neural Network [140]	DoS, Probe, U2R, R2L	96.65, 90.95, 7.59, 10.85	–	–	–	.0125, .0029, .0001, .0038	Acc=91.9%
DT C4.5 and SVM [28]	DoS, Probe, U2R, R2L	–	–	–	–	–	For y=1 DR>99% TT=56.58s TeT=11.20s
KPCA and SVM with GA [141]	DoS, Probe, U2R, R2L	88.69, 98.62, 68, 24.84	–	–	–	–	DR=92.268% FAR=1.025% cc=0.944, TrD=0.944 TeD=1.105
CSVAC [142]	DoS, Probe, U2R, R2L	94.84, 53.25, 44.23, 87	–	–	–	–	DR=95.30% FP=4.25% FN=3.00% TT=3.388s
HNB [143]	DoS, Probe, U2R, R2L	–	–	99.60	–	–	Accuracy=93.72% ER=6.28% Cost=0.2224%
CANN [144]	DoS, Probe, U2R, R2L	–	–	99.99, 99.98, 0, 0	–	–	Overall DR= 99.99% Accuracy=99.76% False Alarm=.003%
CUSUM+sampling technique [157]	DoS	–	–	100 (selective) 92 (sketch-guided)	–	0	–
SVM+LMDRT [158]	DoS, Probe, U2R, R2L	–	–	–	–	–	Overall DR= 99.20% Accuracy=99.31% False Alarm=.60%
LUS-PSO-WMA [104]	DoS, Probe, U2R, R2L	–	–	98.85, 96.85, 99.80, 84.76	–	–	overall acc=92.9016%

A. Single Classifier With All Features

Incorporating single classifiers in IDS was the first step towards intrusion detection using machine learning techniques. Kim and Park [120] proposed a misuse detection approach which applies Support Vector Machine (SVM) for Network Intrusion Detection using KDD'99 dataset. In the initial step, it collects the training and test dataset from KDD'99. The data sets are pre-processed to be used by SVM classifier. SVM is trained over the training dataset and as a result, decision model is generated. This decision model corresponds to hyperplanes

TABLE VI
GLOSSARY OF ACRONYMS USED IN PERFORMANCE MEASURES

Acronyms	Meaning	Description
FP	False Positive	Normal traffic is detected as Intrusive
TP	True Positive	Intrusive traffic is detected as Intrusive (Successful identification of attack)
FN	False Negative	Malicious traffic is detected as Normal
TN	False Positive	Normal traffic is detected as Normal (Successful identification of normal traffic)
ER	Error rate	Misclassification rate of a classifier
DR	Detection Rate	Proportion of correctly classified intrusions to the actual size of attack class
TPR	True Positive Rate	Its equivalent to Detection rate
Acc.	Accuracy	Denotes the total percentage of correctly classified intrusions
TeT$_{/TeD}$	Testing Time	Time taken in testing phase
TT$_{/TrD}$	Training Time	Time taken in Training phase
TDR	Total Detection Rate	Percentage of total attack and normal class correctly detected
NDR	Normal Detection Rate	Percentage of normal class that is correctly detected
ATD	Attack Detection rate	Percentage of attack class that is correctly detected
DDR	DoS Detection Rate	Percentage of DoS attack that is correctly detected
PDR	Probe Detection Rate	Percentage of Probe attack class that is correctly detected
FAR	False Alarm rate	It is the addition of FNR and FPR.
FPR	False Positive Rate	Proportion of incorrectly classified intrusions to the actual size of attack class

in feature space with some support vectors and weight vector values. In the learning process various, C values are taken as 1, 500, 1000 and kernel functions such as linear, 2-poly and Radial Basis Function (RBF). The system tunes the various values of C and kernel function to validate which kernel function is effective and efficient. After learning, validation process is carried out in which test instances are passed to the learned classifier to check for the validity of the classification. The performance is compared in terms of detection rate and misclassification rate. The classifier is not producing good results for detecting Scanning attacks and Low-frequency attacks. The approach is achieving 91.6% detection rate for detection DoS, 36.65% for Probe attack and 12% for U2R attack and 22% for R2L attacks. Researchers have not reported the results for false alarms.

Amor *et al.* [121] performed the Intrusion Detection using two different misuse detection approaches particularly Naive Bayes and Decision tree classifier separately and compared their performance. The KDD'99 dataset is used for training and testing. Decision Tree algorithm builds the tree based on the dataset values. Each nonleaf node corresponds to the test attribute whereas each branch represents the output of the test attribute. An appropriate test attribute is chosen based on the Entropy and Information gain. Leaf node represents the final class of the object. Decision tree produces the rules traversing from root to leaf. In Naive Bayes, conditional probabilities are calculated for each attribute of a test instance corresponding to each class label. The product of this posterior probabilities helps in determining the final class. After the learning phase, the test instances are passed to the classifier to check for the correctness of classification in terms of detection rate. Both are performing very poor in detecting the low-frequency attacks (U2R and R2L). NB achieves detection rate of 96.65% for DoS and 88.33% detection rate for probe whereas DT achieves 97.24% detection rate for DoS and 77.92% detection rate for probe attack. The detection rate is low (0.53%-11.84%) for both approaches as shown in Table V.

Bouzida and Cuppens [122] performed the Intrusion Detection System using Back Propagation Neural Network (BPL NN) classifier and Decision tree separately for misuse detection and compared their performance. They performed experiments on a KDD'99 dataset for training and testing. Before the training of Neural Network, the number of neurons in the input layer is defined as the number of input variables. The number of neurons in the output layer is equal to the total number of classes. They consider only one hidden layer. Neural Network is performing well for detecting DoS and Probe attack, but it fails to detect the low-frequency attacks since the number of records for these attacks is very less in comparison to other attacks (DoS and Probe). To improve the algorithm, an enhanced Decision Tree C4.5 is proposed. In the enhanced algorithm, default condition of original algorithm is treated as new class whereas earlier the default was treated as a normal class. Thus any new instance which does not match the rule will be treated as suspicious. The modified C4.5 algorithm provides improvement for low-frequency attacks too. DT provides the detection rate of 99.99% for DoS, 99.78% for probe, 90.39% for U2R and 98.93% for R2L attacks.

Tajbakhsh *et al.* [131] proposed the misuse detection approach based on fuzzy association rules using KDD'99 dataset. In the training phase, a membership function is used to perform the feature to item transformation. Each attribute-value pair is called an item. The fuzzy membership function is based on Fuzzy C-Means (FCM) clustering algorithm. In the next phase of training, the produced items are reduced. KDD'99 contains 189 items; rule generation over 189 items is not possible. The rules are generated based on the minimum support and confidence values. The fuzzy association rules are used to build the classifier. By the rule sets, an instance is assigned a label. In the testing phase, each feature of a test instance is transformed to an item using the membership function. Then these transformed records are passed through the learned classifier which classifies the instance. The training data sets have been sampled into five sets (normal, DoS, Probe, U2R and R2L). Rules are produced for each class. The total execution time of this classifier is 500s. The technique is not performing well for any of the attacks detection. It provides 78.9% DR for DoS attack detection, 88.5% DR for probe attack, U2R DR for 68.6% and R2L DR for 6.2% attack. The overall detection rate is 70%-90% with 2% false positives.

Kumar and Yadav [140] proposed the simplest model of misuse detection system which is based on Neural Network.

In the first phase, network data set is selected and prepared for training and testing. Further, the selected data is pre-processed to make it compatible with Neural Network by converting all symbolic values into numeric values, for ex. ICMP=1, TCP=2 and UDP=3. After this, the normalization step is performed in which Z-score values are calculated for each feature value. In the next phase, Neural Network is trained over the transformed data set. It consists of three layers input, hidden and output with 41, 29 and 5 neurons respectively. The learned classifier is tested over the testing data set. Neural Network is found performing well for detecting Intrusions except for low-frequency intrusions (U2R and R2L).

Amoli *et al.* [145] proposed an unsupervised clustering based anomaly detection approach which is based on anomaly detection approach to detect and classify the DoS, DDoS, Probe attacks. The model is composed of two detection engines which monitor and inspect the behavior of the network in normal or encrypted communications. The first engine calculates a self-adaptive threshold value to detect the network traffic changes that are caused by attacks such as DoS, DDoS, scanning and worm, etc. The clustering is done in two steps: The network traffic do not pass the threshold, the engine clusters the attack-free traffic according to DBSCAN algorithm. The clustering algorithm calculates the acceptable distance of the network instances and puts the points into the cluster. Once the traffic passes the threshold value, again clusters are created for outliers. The points that cross the acceptable distances are treated as outliers. The second engine aims to detect the botmaster. The first engine sends the IP addresses with attack details to the second engine which then correlates the packets to find the main system controlling DoS. They have considered the ISCX dataset to validate the approach. It achieves 98.39% accuracy, 100% recall, 98.12% precision, 96.39% TNR and 3.61% FPR and outperforms the K-mean outlier detection.

Bhamare *et al.* [152] presented the use of machine learning for detecting attacks in the cyber network. They have executed various machine learning algorithms using two new network datasets other than KDD'99, i.e., UNSW-NB15 and ISOT datasets. These are dynamically generated new datasets which provide real attack statistics. Various misuse detection algorithms such as DT, NB, LR and SVM with three different kernels, which are RBF, Polynomial, Linear. DT provides an accuracy of 88.67%, NB provides 73.8% accuracy, SVM with RBF kernel provides 70.15% accuracy, SVM with polynomial kernel provides 68.06% accuracy, SVM with linear kernel provides 69.54% accuracy, and LR provides 89.26% accuracy. DT provides 6.9% FPR, SVM with RBF function provides 4.1% FPR, SVM with poly function provides 53.3% FPR, SVM with linear function provides 50.7% FPR, NB provides 7.3% FPR, LR provides 4.3% FPR. We can say that among all Logistic regression is providing better results with low FPR. However, the results are not very good using such simple methods of ML.

Jazi *et al.* [157] presented a novel approach for detecting application-layer DDoS attack which uses non-parametric CUSUM algorithm. The authors investigated thirteen sampling techniques to perform filtering on data. Out of them, they observed that selective flow sampling and sketch-guided sampling techniques are performing better and are most effective sampling techniques. The selective sampling provides the best performance with 100% accuracy and no false positive alerts having sampling rate above 20%. A generic sketch-guided sampling also provides good results for detecting application-layer attacks. Sketch guided sampling provides 92% detection rate at 40% sampling rate. Authors have generated various DoS attack traces using different tools and intermixed the attack traffic with the attack-free traffic from ISCX dataset.

Wang *et al.* [158] proposed an intrusion detection framework which uses support vector machine (SVM) integrated with a data transformation method. Feature dependencies are incorporated in the algorithm. Logarithm marginal density ratios transformation (LMDRT) method is used to perform data transformation. The augmented features are of a better quality which is supplied to the SVM. LMDRT method is motivated by Naive Bayes theory for classification as it considers the marginal density ratio. The proposed framework has been evaluated using NSL-KDD 99 dataset. It achieves 99.31% accuracy, 99.20% detection rate and 0.60% false alarm rate.

Various intrusion detection techniques based on single classifier have been discussed. Single classifiers are simple and easy to understand. However, the limitations of single classifier algorithm such as sensitivity towards the choice of input parameters, choice of the kernel function, number of training variables and overfitting, etc. reduce the chances of getting good evaluation results. Secondly, if the dataset has got too many attributes, it becomes difficult for the classifier to provide timely results. Single classifier algorithms, when combined with feature selection algorithms, reduce the computational cost, discussed in next section.

B. Single Classifier With Limited Features

Feature selection techniques are used with single classifier approaches to improve its performance. Sangkatsanee *et al.* [61] proposed a real-time Intrusion Detection System (RT-IDS) based on decision tree C4.5 to detect the two different types of network intrusions such as Denial of Service (DoS) and Probe. The approach is applied in context to misuse detection. The framework consists of three phases: data preprocessing, classification and post-processing. In the preprocessing phase, a packet sniffer is used which uses Jpcap library and network information to extract IP header, TCP header, UDP header and ICMP header, etc. The packet information is considered between the source and destination IP of each packet. Author used Information gain to extract the 12 features as shown in Table VII. Next is classification phase in which the classifier is trained over the training dataset of known labels. In the testing phase, the learned model is used to classify the test instance as normal or intrusive based on the learned behavior. Post-processing is used to reduce the false alarm. In this phase, the network data between source and destination is divided into groups of five records. In each record, if there exist 3-5 records which are reported as the same attack, that group is considered as

TABLE VII
LIST OF IDS TECHNIQUES WITH FEATURE SET

Technique	Features	Feature name
SVM[120]	41	All 41 features
NB vs DT[121]	41	All 41 features
Serial and multiple RBF NN[123]	41	All 41 features
Ensemble of ANNs, SVMs and MARS[124]	41	All 41 features
BPL NN and DT[122]	41	All 41 features
CT SVM[126]	41	All 41 features[126]
PCA[125]	41	All (perform linear transformation on data)
Neuro Fuzzy[147]	41	All 41 features
Active learning based TCM-KNN[127]	8	P34, P35, P38, P5, P6, P10, P23, P13
Flexible NN with PSO and GA[128]	DoS(12)	P1, P8, P10, P11,P16, P17, P20, P12, P23, P28, P29, P31
	Probe (12)	P1, P3, P12, P18, P20, P21, P23, P26, P27, P31, P37, P41
	U2R(8)	P11, P14, P17, P28, P29, P32, P36, P38
	R2L(10)	P1, P3, P11, P12, P13, P18, P20, P22, P25, P38
Bayesian clustering, DT C4.5[129]	41 (1st stage)	All 41 features
	4 (2nd stage)	P1, P3, P5, P6
	14 (3rd stage)	P5, P6, P33, P14, P13, P37, P10, P17, P34, P32, P22, P16, P1, P3
RBF/Elman NN [130]	41	All 41 features
Fuzzy association [131]	NA	All 41 features
FC ANN [132]	41	All
Decision Tree C4.5 [61]	12**	#TCP_Packets, #TCP_src_port, #TCP_destination_port, #Fin_flag, #Syn_flag, #Push_flag, #Ack_flag, #Urgent_flag, #UDP_packets, #UDP_src_ports, #UDP_destination_port, #ICMP_packets
MMIFS with LS SVM [133]	DoS(8)	P2, P3, P5, P6, P23, P24, P36, P41
	Probe (12)	P3, P5, P12, P23, P24, P27, P28, P32, P33, P35, P40, P41
	R2L (14)	P1, P3, P5, P6, P10, P13, P22, P23, P24, P32, P33, P35, P37, P39
	U2L (12)	P1, P2, P3, P4, P5, P6, P14, P21, P23, P24, P32, P33
Hierarchical Clustering and SVM[134]	DoS(19)	P2, P4, P8, P10, P14, P17, P19, P22, P24, P25, P26, P27, P28, P30, P31, P33, P34, P35, P39
	Probe (17)	P3, P4, P12, P22, P23, P24, P25, P26, P27, P29, P30, P31, P34, P36, P37, P39, P40
	R2L (24)	P1, P2, P3, P9, P10, P11, P12, P13, P14, P15, P17, P18, P21, P22, P25, P29, P30, P31, P32, P36, P38, P39, P40, P41
	U2L (24)	P1, P2, P3, P4, P7, P11, P12, P15, P17, P18, P19, P22, P23, P24, P26, P27, P29, P31, P32, P34, P35, P37, P38, P40
FPSO[135]	16	P8, P9, P10, P11, P13, P16, P17, P18, P19, P23, P24, P32, P33, P1, P5, P6
SVM,DT,SA[136]	26	P1, P2, P5, P6, P7, P8, P10, P11, P12, P13, P16, P22, P23, P25, P28, P29, P30, P32, P33, P34, P35, P36, P37, P38, P39, P40
Sub Space Clustering, DBSCAN , EAR[26]	9**	No of source/destination IP adresses, Ports (nSrcs, nDsts, nSrcPorts, nDstPorts), ratio of number of sources to number of destinations, Packer rate (npkts/sec), fraction of ICMP packets (nICMP/npkts) and SYN packets (nSYN/nPkts), avg Packet size (avgPktsSize).
Wrapper (GA, Neuro tree and DT C4.5)[137]	16	NA
Kmean, ACO ,SVM[138]	19	P2, P4, P8, P10, P14, P15, P19, P25, P27, P29, P31, P32, P33, P34, P35, P36, P37, P38, P40
K-mean, Neuro Fuzzy, radial SVM[156]	6	NA
Neural Network [140]	41	All
DT C4.5 and one class SVM (Hybrid)[28]	41	All
KPCA and SVM with GA[141]	NA	Not provided by author
CSVAC[142]	41	All
HNB[143]	7	P3,P5, P6, P19, P23, P31, P37
CANN [144]	6	P7, P9, P11, P18, P21, P20
ANN with (IG+CR) [159]	25	P25 P1,P2,P4,P6,P8,P9,P10,P11,P12,P13,P14,P15,P19, P20,P22,P27,P31,P32,P33, P34, P37, P38, P39, P40, P41
FMIFS [160]	19	P5, P23, P6, P3, P36, P12, P24, P37, P2, P32, P9, P31, P29, P26, P17, P33, P35, P39, P34

All features are described in Table II.
* Features are derived from the corresponding authors

an attack type. The technique is not detecting low-frequency attacks. The detection rate for other attacks is higher than 98% with only two seconds of computational speed. It detects DoS attack with 99.434% DR & 0.73% false alarm and probe attack with 98.868% DR & 0.9% false alarms.

Amiri *et al.* [133] proposed Modified Mutual Information based feature selection approach (MMIFS) and used it with Support Vector Machine (SVM) to detect different types of attacks mainly low-frequency attacks. They have considered training and testing dataset of KDD'99. In the initial phase, data normalization and reduction is carried out by dividing every attribute value by its own maximum value. In the next Phase, feature selection is carried out over the imported training data. MMIFS initially set the feature set empty. It computes the mutual information of the features with the class output and selects the first feature that has the maximum mutual information (MI) value. In the next step, the MI is calculated between features and those features are selected which meets particular criteria which are explained by authors in their work. This step is repeated till the desired number of features are selected. The final set is provided as output to the user. The final set contains 8 features for DoS, 12 features for Probe, 14 features for U2R and 12 features for R2L. The features are shown in Table VII. In the training phase, five SVMs are trained over five different datasets (Normal, DoS, Probe, U2R and R2L). In the testing phase, the attack samples are supplied to each of the trained classifiers which classifies them in one of the five classes. The method does not achieve detection rate greater than 90% to any of the attack detection. Moreover, the detection rate for U2R attack is lowest (30.70%).

Lin *et al.* [144] proposed a new distance based feature extraction approach (CANN) applied with anomaly detection approach based on a k-NN algorithm to detect intrusions. In the first phase, a clustering algorithm is used to make clusters of the training data and then two distances are used to determine the new feature value: First is between a specific data point and its cluster center and second is between a specific data point and its nearest neighbor. New one-dimensional distance based feature value now represents each data point in the training data. In the next phase, principal component analysis (PCA) is used to select the relevant features. Only 6 features are selected as shown in the Table VII. Another phase is classification phase in which k-NN classifier is trained by the new training data set. In the testing phase, CANN process

is again carried out for the test instances and testing data is also represented using one dimensional feature space. k-NN classifier performs the classification over the test data instances. The classifier is not able to detect low-frequency attacks. The overall detection rate is 99.99%, accuracy 99.76% and false alarms 0.003%. It achieves accuracy $\sim$ 99% for both DoS and probe attack detection.

Koc *et al.* [143] proposed the Hidden Naive Bayes Classifier which is an extension of the Naive Bayes classifier for misuse detection. In the first phase of the proposed framework, the attribute values are first converted to discrete values using the entropy minimization discretization and proportional k-interval discretization. In the next phase, feature selection is carried out using three methods: correlation based (CFS), consistency based (CONS) and INTERACT methods. Author has tested the combination of various discretization method and feature selection method with classifier to derive the best method with high detection accuracy. Next phase is the classification phase in which Hidden Naive Bayes classifier is used to learn the behavior of attack sample from training data. Naive Bayes (NB) classifier is based on the assumption of the independent relation of the attributes. Hidden Naive Bayes (HNB) relaxes this assumption and extend the post probability calculation formula which also considers the mutual information of attributes during probability calculation. In the testing phase, a test instance is classified based on the learned behavior. In Intrusion Detection, the attribute values are very much dependent on each other. For example, if we want to check a number of failed logins over a period. Here, content feature (num of failed login) value and basic feature (duration) value both affects the output value and dependent on each other's values too in determining the output. Hence, HNB is improving the NB performance for detecting the intrusions. The author provided DoS detection rate as 99.60% and overall detection as 93.72% using KDD'99 dataset.

Gharaee and Hosseinvand [153] proposed the new feature selection based intrusion detection model (GF-SVM) to detect intrusions in the network. A feature selection approach is proposed where a Genetic algorithm (GA) and SVM are integrated to provide an optimal set of features. Authors have done slight modifications in the fitness function of the GA. Instead of using the accuracy and number of features (NumF) as parameters for fitness function, they have used three parameters: TPR, FPR and NumF. Each parameter is multiplied by certain weight based on the user's choice. In each iteration of GA, each chromosome is evaluated and chromosomes with the highest classification accuracy (using SVM) are selected. The optimal features are used to filter the dataset and Least Squared Support Vector Machine (LSSVM) is used to learn/detect the train/test dataset with selected features. They have considered 7 features for normal attacks and 6-14 features for different types of attacks. The results using UNSW-NB15 dataset are as follows: It achieves an accuracy of 97.45% with 98.47% TPR and 0.04% FPR for detecting normal traffic. It achieves an accuracy of 79.19%-99.45% with TPR 67.31%-100% and FPR 0.01%-0.09% for detecting various types of attacks.

Bamakan *et al.* [161] proposed an intrusion detection framework which integrates the time varying chaos particle swarm optimization (TVCPSO) with multiple criteria linear programming (MCLP) and SVM individually for doing parameter tuning and feature selection. In order to increase the speed of PSO in searching for optimum and avoiding the local optimum, the author introduced the chaotic concept with PSO. The framework has been implemented using NSL-KDD'99. The overall detection rate provided by TVCPSO-MCLP is 97.23% having false alarm rate 2.41% and accuracy 96.88% with feature selection. TVCPSO-SVM provides 97.84% accuracy, 97.03% detection rate and 0.87% false alarm rate with feature selection. Out of the two algorithms, TVCPSO-SVM better results for DoS, Normal and probe and provides detection rate of 98.84%, 99.13% and 89.29% respectively. Whereas TVCPSO-MCLP provides better detection rate for R2L and U2R, having a detection rate of 75.08% and 59.62% respectively.

Akashdeep *et al.* [159] provided an intrusion detection approach which uses ANN and combines it with proposed feature reduction method. All the features of the original dataset are first ranked according to Information Gain method and correlation method individually. Three feature subsets are built using each method, i.e., IG-1, IG2, IG3 and CR1, CR2, CR3. The first subset under each category contains 1- 10 ranked features, second sub-set contains 11-30 features, and the third subset contains remaining features. The first and second subset of each category are combined using union operation & the second subset of each category are combined using the intersection. Rest subsets are ignored. A final subset of features (total 25) is obtained by doing union operation over selected subsets as shown in Figure 9. The reduced KDD 99 dataset with 25 features is used to train ANN classifier. It provides 86.6% detection rate (DR) for U2R, 93.8% DR for DoS, 91.9% DR for R2L and 89.8% DR for probe attack.

Ambusaidi *et al.* [160] proposed flexible mutual information (MI) based feature selection (FMIFS) algorithm which can handle linear and nonlinear features efficiently. FMIFS has been used to select features to remove the most irrelevant features. The filtered dataset is then evaluated by machine-learning based network intrusion detection technique, particularly Least Square Support Vector Machine based IDS (LSSVM-IDS). The performance is measured over the KDD'99 dataset. It provides 99.46% DR, 0.13% FPR and 99.79% accuracy. When evaluated with NSL-KDD'99, it provides 98.76% DR, 0.28% FPR and 99.91% accuracy. It outperforms other methods like MIFS and Flexible Linear Correlation Coefficient Based Feature Selection (FLCFS).

Various intrusion detection techniques based on single classifier with feature selection algorithm, have been discussed. Applying feature selection improves the performance of classification. However, one needs to take care of which combination of feature selection and machine learning algorithm is providing the best results. It also makes the classifier faster over a selected set of attributes of features. However, there is less or moderate improvement in the classification results. The drawbacks associated with one classifier algorithm can be overcome when combined with another classifier algorithm (s) to improve the classification result, discussed in next section in detail.

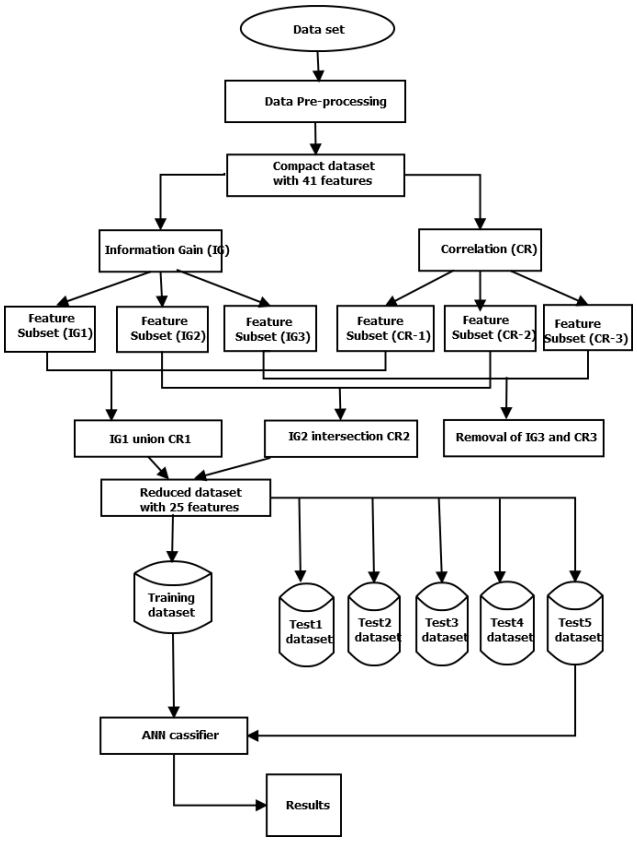

Fig. 9. Feature Reduction technique integrated with ANN.

C. Multiple Classifier With All Features

Kumar and Kumar [146] proposed a multi-objective genetic algorithm (MOGA) for misuse detection. It generates multiple base classifiers and later chooses a subset of classifiers to produce ensemble. It executes in three phases. In first phase, the approach targets to find the optimal Pareto front of non-dominated solutions. A set of base classifiers are produced as candidate solutions. The type of base classifier determines the values of chromosomes and their size. Optimized real values of classifiers are produced as an output of phase 1 which will be used for ensemble. In phase 2, the pareto front of non-dominated solutions (of phase 1) are combined instead of combining the entire population given as input by user. In phase 3, MOGA combines the predictions given by base classifiers to get the final output by ensemble model. They have implemented the proposed approaches using KDD'99 and ISCX 2012 dataset. MLP is used as base classifier. MLP is trained using proposed approach (AMGA2-MLP) and compared with MLP (trained by back propagation method), bagged MLP and boosted MLP. AMGA2-MLP provides better performance than others. For ISCX 2012 dataset, it achieves an average detection rate of 97.0% with 2.4% average FPR. It provides 7% improvement in detection rate and reduction in FPR by 96% over MLP and its ensemble using bagging. The proposed approach can enhance the detection rate by 28% and reduce FPR by 51% over the results of MLP trained by back propagation method using KDD cup dataset.

The next innovation was to integrate many classifiers together to improve the performance of intrusion detection techniques, applying all features of data set. Mukkamala *et al.* [124] proposed the ensemble approach which integrates Artificial Neural Network (ANN), Support Vector Machine (SVM) and Multivariate Adaptive Regression Splines (MARS). Multi-Layer Feed Forward algorithm is very robust in detecting anomalies. However, ANN suffers from the drawback of detecting low-frequency attacks which are in limited numbers in the training data set but ANN requires the huge dataset in training. SVM can perform well even on small data set but it is very slower than other classifiers. MARS is a mathematical process in finding the optimal variable transformations and interactions. It can find the complex data structures too that often hide in high dimensional space. The ensemble of these classifiers is an attempt to reduce the mean squared error and increase the accuracy of classification. In the initial phase, data preprocessor obtains data from the DARPA 1998. In the next phase, each classifier is trained over the data set and individual learned classifiers are formed. In the final phase, majority voting scheme is applied to make the final decision over the test instance. The detected class is one in which majority of the classifiers agreed. The approach is providing good accuracy for each category of attacks.

Zhang *et al.* [123] proposed hierarchical hybrid (misuse and anomaly detection) framework based on multiple Radial Basis Function Neural Networks (RBF NN) and a clustering algorithm in serial and parallel order. In Serial Hierarchical intrusion detection system (SHIDS), the classifier is trained based on the training dataset of KDD'99. In the next phase, traffic is passed through the classifier. Normal packets are verified and passed further but Intrusion packets are detected and saved in the database. A clustering algorithm is used to cluster the attacks based on their statistical distribution. When the number of records in the cluster exceeds some predefined threshold, a new classifier is retrained to the new attack group and added to the last level of classifier. SHIDS suffers from the problem of single point of failure since the classifiers depend on each other. The errors are accumulated in this process. To overcome this problem, author proposed another approach named as Parallel Intrusion Detection System (PHIDS). In PHIDS, anomaly detection classifier is trained in the first level. In second level, misuse detection classier is used to separate the intrusion into four groups based on their signature. In third level, the output of second level is passed to one of the four classifiers which are trained over the separate attack data set such as DoS, Probe, U2R and R2L. These classifiers run in parallel. The detected records are stored in the database. If a number of attacks exceeds the threshold then the corresponding classifier is retrained for the new attacks. Parallel framework with multiple Neural Network classifier works better than single classifiers and achieves around 99% detection rate. RBF can model any nonlinear function using a single hidden layer and requires less computation time in comparison to BPL (Linear Back Propagation) NN.

Toosi and Kahani [147] proposed a Neural Fuzzy classifier by combining the fuzzy logic with the neural network without using any feature selection technique. The proposed misuse detection system consists of two layers. In the first layer, five ANFIS (Adaptive Neuro-Fuzzy Interference System) modules

are trained to observe the intrusive activity. Each classifier is trained over different samples of training data set. Each training sample represents exactly one class (Normal, DoS, Probe, U2R or R2L). Each classifier acts as a signature based classifier. In the next phase, output of ANFIS module is supplied to the fuzzy interference module which maps the output from the Neuro-Fuzzy classifiers to final output space and makes the final decision for normal or intrusive nature. Output is the classifier label of the first layer (neuro-fuzzy output layer) in which the output is the close value to the interference module. Author has also applied Genetic Algorithm (GA) to optimize the structure of fuzzy decision-making system. Author has specified the overall detection rate of the classifier as 95.3% with false positive rate 1.9% over KDD'99 dataset. It has adaptive learning capability. However, it is not working well for low-frequency attacks.

Khan *et al.* [126] proposed the hybrid detection approach by integrating the Support Vector Machine (SVM) with Hierarchical Clustering named as CT SVM using DARPA 1998. Hierarchical clustering is used to reduce the training time of SVM and improve its efficiency of attack detection. SVM uses a hypothesis space of linear function in a higher dimensional feature space. The trained model corresponds to the hyperplanes. The data points closest to the hyperplane are called as support vectors. In the proposed model, hierarchical clustering is performed in the first phase over the training data set using DGSOT algorithm. In each iteration, new nodes are added to the tree based on the learning process. After each iteration, an SVM is trained over the nodes of the tree to reduce the computational overhead. In the next iteration, generated support vectors are passed to the clustering algorithm to control the tree growth. In this way, only support vector nodes grow. The process is continued till some stopping criteria is achieved. The stopping criteria could be based on tree size or tree level or accuracy level. The training time of single SVM is 17.34 hr while integrating it clustering algorithm; it is reduced to 13.18 hr. The system is not working well for detecting low-frequency attacks.

Tong *et al.* [130] presented the Intrusion Detection System which integrates the Radial Basis Function Neural Network (RBF NN) with the Elman Network using DARPA 1998. Neural Network fails to remember the past events. Elman Network overcomes the weakness of Neural Network and provides this capability. It helps in allowing the occasional misuse behavior and detecting the temporally co-located intrusion and collaborative events. Elman Network has a set of the context nodes. Each node takes the input from the hidden node and forwards the output to each hidden node of its hidden layer. Therefore in the hybrid network, both input and hidden node activates the activation node and hidden node fed forward to activate the output nodes. The memorial functionality introduced by context node helps in remembering the past events and correlating the sequence of events. The value of context nodes increases in each iteration and slowly decreases back to zero based on the threshold. The hybrid model is easily adaptable to new intrusions and requires less retraining time. This technique is very helpful in detecting temporarily dispersed and collaborative attacks. The technique is not providing the detection of low-frequency attacks and achieves 100% detection rate for Probe attack.

Wang *et al.* [132] proposed an integrated hybrid intrusion detection approach which consists of Neural Network and Fuzzy Clustering algorithm named as FC-ANN for detecting intrusions. In the first stage, the data set is divided into training and testing phases. Different training subsets are constructed using fuzzy clustering approach. In the second stage, for each training subset an Artificial Neural Network is trained and different ANN classifiers are formulated. The results are combined and a new ANN is trained over the results to reduce the errors. Fuzzy Clustering is performed by using a Fuzzy c-means algorithm which clusters the data points based on the membership grade. The cluster center and membership grades are updated in each iteration to produce the optimized results. ANN is used as classifier to classify the traffic based on the learned behavior of attacks. Sigmoid activation function takes the input as the product of weight and input value and passes the results to other neurons. The output value compared with target and error is calculated. Backpropagation algorithm is used to update the system for new values. The technique produces the overall detection rate as 91.32% using KDD'99. The technique improves the detection rate of ANN for low-frequency attacks.

Feng *et al.* [142] proposed an intrusion detection system (CSVAC) that takes advantage of both misuse and anomaly detection algorithms particularly Support Vector Machine (SVM) and Clustering based Self-Organized Ant Colony Networks (CSOACN). SVM can learn over the little volume of data. CSOACN provides the power of adaptability. In the real-time Intrusion Detection System, it is essential that whenever new data points are added, the old model should be updated immediately. It saves a significant retraining time of the system. In the first phase, the training data is normalized to remove the biasness of some features over others. Next phase is training phase in which SVM is trained over the several training data subsets repeatedly. In the first iteration, initial hyperplanes are generated randomly. After that in each iteration, CSOACN clustering algorithm selects the points around the generated support vectors and forms clusters. These data points will be used for training the SVM in the next iteration. Here, CSOACN not only learns the data points and identify the outliers but also makes the data selection for training the SVM. At the end of the training phase, we have two learned classifiers: SVM and CSOACN. In the testing phase, the test instances are passed through the classifiers. If both classifiers classify instance as anomalous, then only it is flagged as anomalous. CSOACN is used to determine the subclass of the attack. If results differ from each other, data item is labeled as "amphibolous". It can be further used for analyzing the behavior of normal or intrusive data. The classifier provides 3.388 s of training time, but it is not performing well for Probe and U2R attack detection.

Elhag *et al.* [148] combined the genetic fuzzy system (GFS) with the pairwise learning (one to one mapping: OVO) architecture. The use of fuzzy sets creates a smoother borderline between rules set and pair-wise learning improves the precision of the rare attack patterns. The combined misuse

detection approach provides the high performance and is being compared with decision tree by the authors. In GFS, fuzzy association rules (FARC-HD) form the base classifier. The inner working of the FARC-HD applies the genetic algorithm to optimize the membership values of FARC-HD and obtain the compact rules. The OVO method converts the multiclass classification problem into binary sub-problems by making all possible pair of classes. Then a binary classifier is trained for the subsample of data ignoring other samples that do not belong to its related class. Each of the trained models processes an instance and then the predictions by all classifiers are combined to obtain the output. They have used preference relations solved by non-dominance criteria to combine the results. They have selected 5 labels per variable for fuzzy sets. The minimum support considered is 0.05 and minimum confidence is 0.8. The approach achieves 99% overall accuracy with 97.77% attack detection rate and 0.191% false alarms. The accuracy is 98.05% for DoS attack, 95.83% for probe attack, 87.54% for R2L and 65.38% for U2R. The approach is achieving the high accuracy for R2L and U2L and outperforming the layered approach discussed above.

Yassin *et al.* [149] proposed the hybrid detection approach which provides integration of K-Means clustering and Naive Bayes classifier. The combination of anomaly detection and misuse detection is used to detect the attacks which can be bypassed by having only one type of detection mechanism. In the first phase, K-means clustering is used as a pre-classification module to make the clusters. Each cluster represents the group of similar data. The entire data is labeled with Kth cluster set. Afterwards, Naive Bayes algorithm is used as a classification module to classify the data instances of the labeled cluster into the attack and normal. The approach is implemented using ISCX 2012 attack dataset. It achieves an accuracy of 99.8% with 95.4% detection rate and 0.13% false alarms. The integration improves the accuracy of Naive Bayes which provides 82.8% detection rate and 17.6% false alarms.

Various intrusion detection approaches based on multiclassifier algorithm are discussed. The classifiers are trained to learn all features of the training dataset. There is an improvement in the accuracy of the system. However, computational cost and complexity of the system are high. The detection rate is improved by multiple classifier algorithms especially for low-frequency attacks such as U2R and R2L. Combining a bunch of classifiers may not always work better. Various possible different combinations of multiclassifier needs to be cross-validated. The performance and speed can be improved by integrating the multiple classifier techniques with suitable feature selection approach, discussed in next section.

D. Multiple Classifier With Limited Features

Feature selection techniques are further used with Hybrid Classifiers for further improvement, especially for low-frequency attack detection. Chen *et al.* [128] proposed a misuse detection approach based on flexible neural Tree (FNT) technique using DARPA 1998. The parameters of FNT are optimized using the particle swarm optimization technique. Genetic Algorithm is used to select the features for the algorithm. 12 feature for DoS, 12 features for Probe, 8 features for U2R and 10 features for R2L are selected using 41 variable input data set. The working of the flexible neural tree is based on the neural network except it is flexible to expand. The tree expands base on the fitness function. Particle Swarm Optimization (PSO) technique is used to optimize the parameters during this process. PSO conducts a search using a population of particles that corresponds to an individual in an evolutionary algorithm. The algorithm is working well for all categories of attack detection.

Xiang *et al.* [129] proposed a hybrid detection algorithm based on Decision Tree C4.5 and Bayesian (AutoClass) clustering algorithm using KDD'99 dataset. The technique operates in four stages. In the first stage, Decision Tree C4.5 is used to classify the training data set into three categories (DoS, Probe and others). Decision Tree fails to separate the U2R and R2L attacks. In the second stage, Bayesian Clustering is used to separate the normal connections from U2R and R2L connections. Clustering algorithm performs better than supervised algorithms in detecting the low-frequency attacks. In this stage, four features are used for clustering: duration, service, src bytes and dst bytes. In this phase 178 clustered are formed and 31 are declared as attacks. In the third phase, again decision tree C4.5 is used to separate the U2R from R2L attacks. This is easier since normal connections are filtered out in the second stage. In this stage, only 41 features are used as shown in Table II (Section III) and Table VII (Specific features). The last stage further specifies individual U2R and R2L attacks based on the given training data. This classification is well effective for known attacks and results depends on the availability of sufficient label data. This technique is performing low for R2L attacks detection.

Lin *et al.* [136] proposed an algorithm that uses multiple machine learning algorithms to detect the intrusions namely Support Vector Machine (SVM), Decision Tree (DT) and Simulated Annealing (SA) in context of misuse detection. It takes advantage of all three classifiers such as SVM performs well for classifying the intrusions, DT can produce rules and SA coverage to global optima. In the first phase, KDD'99 dataset is prepared for training and testing purpose. In the next stage, SVM and SA are combined to select the best features. SVM maps the training data into high dimensional feature space. SVM is trained and tested with the different possible feature set and at last best feature set is selected with maximum accuracy. During this process, SA is used to optimize the two of the parameters (C and λ) used by SVM with Gaussian radial basis function kernel. Here C is the parameter for the soft margin cost function, which controls the influence of each support vector. λ is the free parameter of the Gaussian radial basis function. In the next phase, DT is used to produce the rules for the classification using the selected features of the previous phase. Information gain and Entropy are two important measures used by this algorithm while building the decision tree. Here, SA is again used to optimize the two parameters of DT: pruning confidence factor (CF) and minimum cases (M). This technique is performing well for detecting all types of attacks. Especially for DoS, the detection rate is 100%.

Casas *et al.* [26] proposed Unsupervised Network Intrusion Detection System (UNIDS) which uses three algorithms Sub-Space Clustering, DBSCAN and Evidence Accumulation for Ranking Outliers (EA4RO) for anomaly detection. Sub Space clustering is used to project the X (a feature vector with n dimensions) into N k-dimensional Sub Spaces (X_i). Each Sub Space X_i is then partitioned using DBSCAN clustering algorithm. DBSCAN algorithm generates clusters of various densities. Anomalies present in low-density areas. It output the set of clusters with a set of outliers. Clustering in low dimensional space is much faster and efficient. Outliers represent the different IP flows. EA4RO algorithm is used to rank these flow based on their degree of abnormality (dissimilarity). The degree of dissimilarity is measured as a distance from outlier to the centroid of the biggest cluster. Here Mahalanobis distance measure is considered to measure the dissimilarity. The flows are ranked according to their dissimilarity measure. IP flows whose dissimilarity is greater than a predefined threshold are marked as anomalous. The algorithm is working well detecting all attacks. Especially for Probe attacks, accuracy is 100% when tested with the KDD'99 dataset. It is using 9 features as shown in Table VII.

Li *et al.* [138] proposed a hybrid approach based on Support Vector Machine (SVM), ant colony algorithm and clustering algorithm. There are three phases of classification. In the first phase, the training data set (KDD'99) is pre-processed by deleting the repeated data in the database. Data compaction is further achieved by using the K-mean Clustering which groups data items in different clusters based on their similarity measure. The intersection of original data and clusters remain. It reduces the size of data and makes it more efficient for classification. SVM takes very long time to process the huge database. In the next phase, training subsets are selected using the ant colony algorithm. The effectiveness of each subset is evaluated using the SVM classifier. In the third phase, feature selection is performed using proposed Gradual Feature Removal (GFR) method. In GFR method, a feature is removed from the feature set one by one and accuracy of the classifier is examined for the feature set. The process is carried out for each feature in the feature set and influence of a feature is noted. The most balanced feature set is selected based on the accuracy of the classifier. In the next phase, SVM learns the behavior of attack data supplied to it based on the selected feature set. The proposed technique does not perform well for detecting U2R attacks. However, the detection rate is greater than 90% for other attacks.

Kuang *et al.* [141] proposed the misuse detection system based on Kernel Principal Component Analysis (KPCA), Support Vector Machine (SVM) and Genetic Algorithm (GA). SVM provides the good generalization capability over small sample training data. GA performs the feature selection for classification. In the first phase, data preprocessing is carried out over training dataset (KDD'99) to transform all data values into the numeric form and normalize all values. Training data is divided into five subclasses based on the category. In the next phase, KPCA transfers the high dimensional feature space into a low dimensional eigenspace. Then GA performs the feature selection by iteratively selecting the populations and applying selection, crossover, and mutation to find the optimal subset till stop criteria is reached. In each iteration, the effectiveness of the feature set is determined by the classifier's accuracy. Further different SVM classifiers are trained over the selected parameters over the five training data subsets. The overall detection rate is 92.268% with 1.025% false positive rate. However, the technique is not working well for low-frequency attacks.

Chandrasekhar and Raghuveer [139] proposed a hybrid Model which uses the power of Clustering (K-means), Fuzzy Neural Network (neuro-fuzzy) and Support Vector Machine (SVM) to identify the intrusions. Processing the huge chunk of data introduces errors and affects the efficiency of the classifier. Hence, in the initial phase, the proposed framework divides the training data set into small subsets based on the similarity of the data items by using K-means clustering algorithm. It reduces the sparsity in data and makes it more suitable for the classifier. In the next phase, five neuro-fuzzy classifiers are trained over the five training subsets. It is difficult to determine the number of neurons and hidden layers in the Neural Network. The problem is overcome by introducing the fuzzy logic with the neural network. It can manage imprecise, partial and vague information. It uses the backpropagation algorithm to find out the input membership function. Each Neuro-Fuzzy Network outputs the set of features with the membership value. In the next phase, SVM is trained using the selected features for each of the training samples and support vectors are generated. In the testing phase, SVM classifies the test instances based on the generated hyperplanes. The algorithm is performing well for detecting all types of attacks.

Horng *et al.* [134] proposed a hybrid framework based on hierarchical clustering and Support Vector Machine (SVM) using KDD'99 dataset. To improve the detection rate of SVM and to reduce its training time, Clustering is integrated with SVM. In the first phase, data transformation and scaling is performed to convert the non-continuous values into continuous form and normalize the values on a scale [0-1]. In the next phase, Clustering Feature (CF) tree is constructed for each category of attack. A CF tree is a balanced height tree with two parameters: branching factor (B) and radius threshold (T). It is compact representation of the dataset. Insertion and deletion of a new data point are same as B+ tree. One entry in the leaf node represents one cluster. The hierarchical clustering organizes the training data in tree form, making the data more balanced. In the next phase, feature selection is carried out using the gradual feature removal method, in which each attribute is taken out once and accuracy of the classifier is noticed to examine the influence of the feature in the output. The output of this phase is 19, 17, 24 and 24 features for DoS, Probe, U2R and R2L respectively. In the next phase, SVM is trained over the selected features using RBF (Radial Basis Kernel Function). In testing, a test instance is classified as per the learned behavior of classifier. The system achieves 95.72% accuracy with 0.7% false positive rate. The performance is good for DoS and Probe and poor for low-frequency attacks.

Gupta *et al.* [150] proposed a layered misuse detection approach for achieving high accuracy and high efficiency. The attack accuracy is achieved by using the Conditional Random

Fields (CRF) and high efficiency is achieved by using the layered approach. CRFs are probabilistic systems which are used to model the conditional distribution over a set of random variable. CRFs have some advantages over Markov models as they are undirected models used for sequence tagging which makes them free from the observation bias and label bias. They have considered four attack groups namely, DoS, Probe, U2R and R2L namely. They have selected features separately for each class based upon the attack characteristics without using standard feature selection algorithm. Four different training sets are created for each of the layers with comprises one of the attack class and normal class. Each layer is trained for specific attack category using specific feature set for the attack class. In each iteration of the algorithm in training phase, a CRF model is trained for each layer for a specific class. After training, there are four CRF models which are plug-in sequentially in such way that the connections labeled as normal are passed to the next layer otherwise detected as attack class (corresponding to the layer) and connection is blocked. The layered approach achieves 98.60% detection rate for probe with 0.91% false alarms, 97.40% detection rate for DoS attack with 0.07% false alarms, 86.3% detection rate for U2R with 0.05 false alarms and 29.60% detection rate for R2L with 0.350% false alarms.

Mamun *et al.* [151] proposed a deep packet inspection technique based on the use of Shannon's entropy to identify the application flows, part of encrypted traffic. The feature set is composed of the following features: the entropy of the entire payloads, sliding window or n-gram length, the entropy of encoded payload and Bi-Entropy, considered for both binary payload and encoded payload. A logarithmic function is used to transform all the metrics as it was found to improve the accuracy of the classifier. All the features are further preprocessed using genetic algorithm. The genetic algorithm is integrated with the Least Square SVM (LSSVM), used as a training algorithm. In each iteration, a genetic algorithm is used to select the features based on the fitness function. The fitness function is calculated by weighting the true positive rate, false positive rate and a total number of selected features. Total 10 features are considered using GF-LSSVM. LSSVM is trained with the selected features to classify the traffic. It achieves the detection rate of 96.7% for encrypted traffic and 96.6% for unencrypted traffic with almost similar false alarm $\sim$ 0.03% using ISCX dataset.

Chowdhury *et al.* [154] provided the use of machine learning for network intrusion detection. They have applied the combination of simulated annealing (SA) [162] and Support Vector Machine (SVM) [20] to improve the detection accuracy and reduce the false alarms. Misuse detection algorithms have the power to classify the normal and abnormal classes, given the attack behavior. In the proposed misuse detection algorithm, first n features are selected using the SA algorithm from a set of K features. Now dataset N with n selected features is used to train the SVM. The trained model is used to detect the future test instances. The experiments have been performed on the UNSW-NB dataset. From the dataset, 150, 000 samples are selected randomly which contains 75,000 normal and 75,000 anomaly samples. The 70% of the total dataset is used for training and 30% is used for testing. The normal SVM provides an accuracy of 88.03% whereas the proposed scheme provides 98.76% accuracy with a randomly selected three features using SA approach. It provides 0.09% FPR and 1.35% FNR rate which is reasonably low.

Moustafa and Slay [155] proposed a hybrid feature selection approach to reduce the irrelevant set of features and to integrate it with other machine learning algorithms for intrusion detection. The proposed NIDS architecture makes use of both anomaly and misuse detection approaches for intrusion detection. NIDS, first of all, takes the input from the dataset (UNSW-NB15 or NSL KDD'99). It then calculates the center points for attribute values. A center point or mode is the most frequent value of the attribute. The center points for all the features are used as an input to the association rule mining algorithm (Apriori) to reduce its processing time. The association rule mining finds out the correlation of the two or more features/attributes and finds out the highly ranked features. The dataset is filtered based on the selected features and given as an input to the detection engine. Here, three algorithms namely Expectation-Maximization (EM) clustering, Naive Bayes (NB) and Logistic Regression (LR) have been used. The results using UNSW-NB15 are as follows: EM provides an accuracy of 77.2% with 13.1% FAR. LR provides accuracy of 83.0% with 14.2% FAR and NB provides 79.5% accuracy with 23.5% FAR.

Aburomman and Reaz [104] proposed an ensemble ML based intrusion detection approach in which six k-NN and six SVM classifiers are trained over KDD'99. The results of all the 12 trained models are combined using three different approaches. In first way, PSO generates weights which are combined by Weighted Majority Voting algorithm (WMA) to combine the results of trained models. In second way, the behavior parameters of PSO are combined using Local unimodal sampling (LUS) and rest is same as the first approach. In third way, WMA is used to fuse the results of classifiers. In all three ways, the results are produced and compared. LUS-PSO-WMA (second way of combining results) provides better accuracy among all. It provides 83.6878% accuracy for detecting normal traces, 96.8576% accuracy for probe attacks, 98.8534% for DoS attacks, 99.8029% for U2R attacks and 84.7615% for R2L attacks as shown in Table V.

Various intrusion detection techniques based on multiple classifier algorithms which are integrated with suitable feature selection approach have been discussed. Applying feature selection, definitely improves the speed of classification and in some cases, it is improving the detection rate also. However, time complexity is not much reduced. The overall complexity is still high as it consists of multiple classifier and feature selection algorithms. This class of algorithms can make use of parallel programming techniques to reduce the training time. Also, obtaining good classification results for low-frequency attacks is still a challenge.

VI. Classification of Techniques for a Specific Attack Detection

In this Section, we provide the classification of various techniques for different attacks based on their performances. It helps the readers in choosing a particular technique for

TABLE VIII
CLASSIFICATION OF TECHNIQUES FOR DOS ATTACKS

Denial of Service Attacks			
Single Classifier with 41 features (Technique Name, Features, Detection Rate)	**Single Classifier with limited features**(Technique Name, Features, Detection Rate)	**Multiple Classifier with 41 features**(Technique Name, Features, Detection Rate)	**Multiple Classifier with limited features**(Technique Name, Features, Detection Rate)
• **Decision Tree, 41, 97.24% [121]** • Neural Network, 41, 97.00% [122] • Naive Bayes, 41 96.65% [121] • SVM, 41, 91.6% [120] • Fuzzy Association, 41, 78.9% [131]	• **CANN, 6, 99.99% [144]** • Decision Tree, 12, 99.43% [61] • Hidden Naive Bayes, 7, 99.6% [143] • MMIFS with SVM, 8, 78.69% [133]	• **(Ensemble of ANN, SVM, MARS), 41, 99.97% [124]** • FC-ANN, 41, 99.91% [132] • Neuro Fuzzy, 41, 99.5% [147] • Multi NN, 41, 99% (Specific smurf) [123] • CT SVM, 41, 97.35%. [126] • CSVAC, 41, 94.84% [142] • ANN with Elman Network, 41, 87.5% [130]	• **SVM, DT & SA, 23, 100% [136]** • Hierarchical Clustering & SVM, 19, 99.5% [134] • FNT, PSO & GA, 12, 98.75% [128] • Bayesian Clustering & DT, (41, 4, 14), 98.66% [129] • Subspace Clustering, DBSCAN, EAR, 9, 98% [26] • K-mean, ACO, SVM, 19, 97.67% [138] • K-mean, Neuro fuzzy, SVM, 6, 97.64% [139] • FPSO, 16, 97.22% [135] • (KPCA, SVM, GA), NA, 99.69%[141]

detecting the specific attack. The detailed description of these techniques is presented earlier in Section V, and the features with techniques are also shown in Table VII. Most of the approaches use KDD'99 as a common dataset for evaluation. Hence, these are considered for comparison.

A. Detection of Denial of Service Attacks

Single classifier approach is much easier to implement. Decision Tree (DT) is performing well in this category with detection rate of 97.24%. However, the detection rate is improved to 99.43% when applied with 12 features. CANN [144] is achieving 99.99% accuracy with 6 features. However, it will perform very well in detecting the Land attack. For other DoS attacks, it needs to improve its feature set (the reason is explained in summary of observations in Section VII-B). ANN when combined with SVM and MARS [124], improves its performance and provides 99.97% classification rate. The performance of ANN [140] is also improved when combined with Fuzzy Clustering (FC-ANN) [132] with 99.91% detection rate. Fuzzy Logic with ANN (FC-ANN) is providing 99.5% detection rate. Now if we talk about SVM [120], its detection rate is also improved when combining it with Clustering approach (CT SVM) [126]. Earlier the detection rate was just 91.6% and later it improved to 97.35%. In fact, there is also an improvement if we combine SVM with ant colony networks (CSVAC) [142] and detection rate improves to 94.84%. All the above mentioned hybrid techniques are considering 41 features which will affect the computation time. SVM in combination with DT and SA [136] is proving the highest detection rate of 100% with 23 features and when combined with Hierarchical clustering it achieves 99.5% detection with 19 features. The techniques are mainly using KDD'99 data set. The classification of various other techniques with detection rate for DoS attack detection is shown in Table VIII.

B. Detection of Scanning (Probe) Attacks

Single classifiers with all 41 features are not performing well for Probe attacks detection. None of the classifiers is achieving 90% in this category. Decision Tree [121] with 41 features is providing detection rate of just 77.92% but when applied with selected features (12 features) [61], the detection rate improved to 98.868%. Similarly, SVM [120] provides very poor detection rate of 36.65%, but when MMIFS [133] feature selection technique is applied which uses only 8 features, its detection rate is improved to 86.46%. Further integrating the single classifiers provides a significant improvement in detection. ANN [122] is achieving 71.63% detection rate but when it is integrated with Elman network [130] it provides 100% detection. Even the integration with SVM and MARS [124] is also achieving 99.85% detection rate. Results are further improved in terms of detection rate and computational time with Hybrid classifiers with feature selection. Here the combination of Subspace clustering, DBSCAN and EAR [26] algorithm is achieving 100% detection rate with 9 features. FNT technique with PSO and GA [128] also achieves 98.39% detection rate with 12 features. SVM integrated with DT and SA [136] is performing far better with a detection rate of 98.35% with 23 features in comparison to only SVM technique (36.65%) with 41 features. The performance of techniques is shown in Table IX.

C. Detection of User to Root Attacks

We have classified techniques based on their detection rate for U2R attacks. We have discussed those techniques which are performing well.

In case of U2R attacks, the performance of the single classifier is very poor. However, the integration with other classifiers is improving their performance. FNT approach with PSO and GA [128] performs very well and achieves the highest detection rate of 99.7% with 12 features (refer Table VII for features in Section V). Intrusion Detection technique which uses Multiple Neural Network classifiers [123] is providing 99.7% detection rate for a specific type of U2R attack, i.e., guess password. There is a significant improvement of Neural Network [140] performance as its detection rate is improved to 93.18% when it is combined with the fuzzy clustering

TABLE IX
CLASSIFICATION OF TECHNIQUES FOR SCANNING ATTACKS

Scanning Attacks			
Single Classifier with 41 features (Technique Name, Features, Detection Rate)	**Single Classifier with limited features**(Technique Name, Features, Detection Rate)	**Multiple Classifier with 41 features**(Technique Name, Features, Detection Rate)	**Multiple Classifier with limited features**(Technique Name, Features, Features, Detection Rate)
• Naive Bayes, 41, 88.83% [121] • Fuzzy Association, 41, 88.50% [131] • Decision Tree, 41, 77.92% [121] • Neural Network, 41, 71.63% [122] • SVM, 41, 36.65% [120]	• **CANN Approach, 6, 99.98% [144]** • Decision Tree, 12, 98.868% [61] • Hidden Naive Bayes, 7, NA [143] • MMIFS with SVM, 8, 86.46% [133]	• **ANN with Elman Network, 41, 100% [130]** • (Ensemble of ANN, SVM, MARS), 41, 99.85% [124] • Multi NN, 41, 99.5% (Specific ipsweep)[123] • CT SVM, 41, 91.2% [126] • Neuro Fuzzy, 41, 84.1% [147] • CSVAC, 41, 53.25% [142] • FC-ANN, 41, 48.12% [132]	• **Subspace Clustering, DBSCAN, EAR, 9, 100% [26]** • (KPCA, SVM, GA), NA, 98.62% [141] • FNT, PSO & GA, 12, 98.39% [128] • SVM, DT & SA, 23, 98.35% [136] • Hierarchical Clustering & SVM, 19, 97.5% [134] • Bayesian Clustering & DT, (41, 4, 14), 93.40% [129] • K-mean, ACO, SVM, 19, 91.45% [138] • K-mean, Neuro fuzzy, SVM, 6, 86.7% [139] • FPSO, 16, 77.77% [135]

TABLE X
CLASSIFICATION OF TECHNIQUES FOR U2R ATTACKS

User to Root Attacks			
Single Classifier with 41 features (Technique Name, Data Set, Detection Rate)	**Single Classifier with limited features**(Technique Name, Features, Data Set Detection Rate)	**Multiple Classifier with 41 features**(Technique Name, Data Set, Detection Rate)	**Multiple Classifier with limited features**(Technique Name, Features, Data Set Detection Rate)
• **Fuzzy Association, 41, 68.60% [131]** • SVM, 41, 12% [120] • Decision Tree, 41, 13.60% [121] • Naive Bayes, 41, 11.84% [121] • Neural Network, 41, 0% [122]	• **MMIFS with SVM, 8, 30.70% [133]** • CANN Approach, 6, 0% [144] • Decision Tree, 12, 0% [61] • Hidden Naive Bayes, 7, NA [143]	• **Multi NN, KDD'99 (99.7%) (Specific guess password)[123]** • FC-ANN, KDD'99 (93.18%)[132] • Ensemble of ANN, SVM, MARS, DARPA 98 (76%)[124] • CSVAC, KDD'99 (44.23%)[142] • Neuro Fuzzy, KDD'99 (41.1%)[147] • CT SVM, DARPA 98 (17.23%)[126] • ANN with Elman Network, DARPA 98 (0%)[130]	• **FNT, PSO & GA, 12 DARPA 98 (99.7%)[130]** • Subspace Clustering, DBSCAN, EAR, 9, KDD'99 (88%)[26] • Bayesian Clustering & DT, (41, 4, 14), KDD'99 (71.43%)[129] • FPSO, 16, KDD'99 (69.44%)[135] • KPCA, SVM, GA, NA, KDD'99 (68%)[141] • SVM, DT & SA, 23, KDD'99 (80%)[136] • K-mean, ACO, SVM, 19, KDD'99 (53.84%)[138] • Hierarchical Clustering & SVM, 19, KDD'99 (19.7%)[134] • K-mean, Neuro fuzzy, SVM, 6, KDD'99 (3.83%)[139]

approach (FC-ANN) with no change in the set of features. Earlier the detection rate of ANN was 0% for U2R attacks. The Integration of ANN with SVM and MARS [124] is also proving an improvement with a detection rate of 76%, but it is not acceptable. There is very little improvement in SVM [120] (12%) when it is integrated with clustering approach (CT SVM, 17.23%) [126] which is also not acceptable. Hybrid techniques with feature selection are providing good detection rate for U2R attacks. The performance of other techniques is shown in Table X.

D. Detection of Remote to User Attacks

We have classified techniques based on their detection rate for R2L attacks. The techniques are discussed which are performing well.

In case of R2L attacks, the performance of the single classifier is very poor. However, the integration with other classifiers is improving their performance. Here ensemble of SVM, ANN and MARS [124] is proving 100% detection rate as claimed by the author which is a significant improvement over ANN [122] (26.68% detection rate). However, they have not mentioned anything about what specific R2L attacks are detected. A hybrid classifier with feature selection is also performing very good. Here FNT approach with PSO and GA [128] is providing detection rate of 99.09% with 12 features. FPSO is achieving the detection rate of 97.22% with 16 features. Integration of SVM, DT and SA [136] is achieving a detection rate of 90.67% with 23 features. The performance of other techniques is shown in Table XI.

Multiple classifiers with feature selection are performing better for R2L than U2R attacks detection. This is because of (1) Most of the hybrid classifiers are filtering the data before training the classifier by performing clustering on the data which group data items into groups

TABLE XI
CLASSIFICATION OF TECHNIQUES FOR R2L ATTACKS

Remote to Local Attacks			
Single Classifier with 41 features (Technique Name, Data Set, Detection Rate)	**Single Classifier with limited features**(Technique Name, Features, Data Set Detection Rate)	**Multiple Classifier with 41 features**(Technique Name, Data Set, Detection Rate)	**Multiple Classifier with limited features**(Technique Name, Features, Data Set Detection Rate)
• **Neural Network, KDD'99 (26.68%)[122]** • SVM, KDD'99 (22%)[120] • Naive Bayes, KDD'99 (8.64%)[121] • Fuzzy Association, KDD'99 (6.2%)[131] • Decision Tree, KDD'99, (0.52%)[121]	• **MMIFS with SVM, 8, KDD'99 (84.85%)[133]** • CANN Approach, 6, KDD'99 (0%)[144] • Decision Tree, 12, RLD09 (0%)[61] • Hidden Naive Bayes, 7, KDD'99 (NA)[143]	• **Ensemble of ANN, SVM, MARS, DARPA 98(100%)[123]** • Multi NN, KDD'99 (98.8%) (Specific buffer overflow)[123] • CSVAC, KDD'99 (87%)[142] • FC-ANN, KDD'99 (83.33%)[132] • CT SVM, DARPA 98 (43.98%)[126] • Neuro Fuzzy, KDD'99 (31.5%)[147] • ANN with Elman Network, DARPA 98 (0%)[130]	• **FNT, PSO & GA, 12 DARPA 98(99.09%)[128]** • FPSO, 16, KDD'99 (97.22%)[135] • SVM, DT & SA, 23, KDD'99 (90.67%)[136] • K-mean, ACO, SVM, 19, KDD'99 (90.34%)[138] • Subspace Clustering, DBSCAN, EAR, 9, KDD'99 (85%)[26] • Bayesian Clustering & DT, (41, 4, 14), KDD'99 (46.96%)[129] • Hierarchical Clustering & SVM, 19, KDD'99 (28.8%)[134] • KPCA, SVM, GA, NA, KDD'99 (24.84%)[141] • K-mean, Neuro fuzzy, SVM, 6, KDD'99 (7.661%)[139]

based on their similarity measure. It brings the balance in data which is very crucial in machine learning algorithms. (2) Multiple classifiers are more adaptable than single classifiers and hence can learn the new attack behavior efficiently. However, still, there are various difficulties associated with detecting low-frequency attacks as discusses in Section VIII.

Existing intrusion detection approaches based on machine learning have been thoroughly analyzed with respect to individual attack categories. Limitations associated with approaches for each category are discussed with viable solutions. No one particular machine learning algorithm can help in detecting all types of attacks. Hence, the use of a specific algorithm (misuse, anomaly or hybrid) is recommended for detecting a specific set of attacks.

VII. PERFORMANCE ANALYSIS OF DIFFERENT MACHINE LEARNING ALGORITHMS IN INTRUSIONS DETECTION

We have carried out the critical performance analysis of various machine learning techniques in detecting all types of attacks. The analysis is carried out with respect to each category of machine learning techniques which are earlier described in Section V. The results, reported by researchers have been analyzed and compared. Based on the observation, we found that most of the work has been validated using KDD'99. Hence, the performance analysis of techniques based on KDD'99 has been carried out. The best performing techniques have been discussed for each attack category and also the limitations of each category of techniques and the solutions applied to overcome the limitations are provided. We provide the overall conclusion at the end of each category. It provides readers with a clear view of limitations in intrusion detection techniques and why the integration and feature selection is applied.

A. Performance Analysis of Single Classifier Algorithms With 41 Features

Among all four Classifications, single classifiers with all features are performing very low for the detection of various attacks. We have mainly considered five standard machine learning classifiers: Decision Tree (C4.5) [121], Neural Network [140], Naive Bayes [121], Support Vector Machine [120] and Fuzzy Association rules [131] as shown in Figure 10. Decision tree [121] gives better detection rate (97.24%) for DoS attacks compared to other four classifiers. Neural Network [140] achieves highest detection rate (90.95%) for Probe attacks whereas Fuzzy Association rules [131] provides far better results for U2R attacks with detection rate 68.6% as it uses data reduction technique. However, Fuzzy rules are not working well for DoS attacks Detection. It meets only 78.9% Detection rate. SVM [120] Classifier has highest detection rate around 22% for U2R attacks which is not acceptable in an environment where security is an important aspect. Although KDD'99 contains a large number of DoS and Probe connection records, even then single classifiers are not able to detect the behavior of these attacks. The reasons for the degrading performance can be explained in terms of two major aspects.

(i) Low Detection Rate & High Computational Cost: Classification task becomes time-consuming and hectic when considering all features of data. It results in increasing computation power, storage and error rate and it affects the performance of classifier badly. The classifier suffers from the problem of "Curse of Dimensionality". The problem can be explained as when the dimensionality increases, the volume of the space increases so fast that the available data becomes sparse. This sparsity could be problematic for machine learning algorithms because of statistical significance of training dataset. In high dimensional data, all objects appear to be sparse and dissimilar in many ways; data organizing strategies may not work efficiently

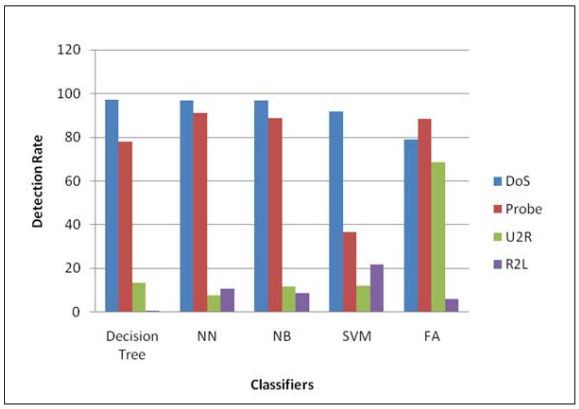

Fig. 10. Performance comparison of single classifiers with 41 features for different Attacks.

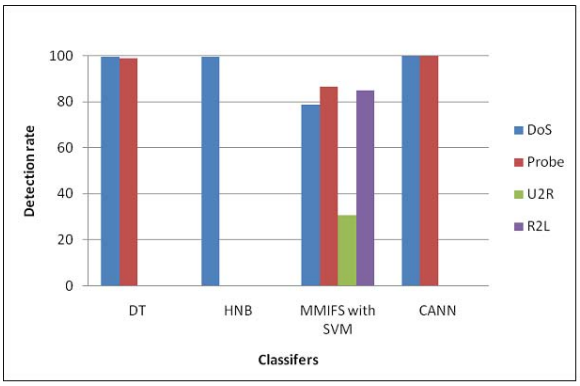

Fig. 11. Performance comparison of single classifiers with feature selection.

in such a sparse data organization. Hence there is a dire need to find optimal feature for analyzing the behavior of an attack and making a classifier learn the behavior efficiently and accurately. How many features does the algorithm should employ is one of the most important aspect that a researcher must take into consideration before performing the classification.

(ii) Algorithmic Drawbacks: The algorithmic drawbacks of each classifier is being discussed in details in Section IV-A.

The above limitations of the single classifiers with all features make them less suitable for attack detection. Some problems are computational complexity, less adaptability, sensitivity towards input change, sensitivity towards the choice of the kernel function and its parameters, number of training variables, algorithmic complexity and overfitting. This causes average detection rate for high-frequency attacks (DoS and Probe) and poor detection rate for low-frequency attacks (U2R, R2L).

Various **solutions** for overcoming the limitations of single classifiers are as follows:

i) Incorporating Feature Selection/Feature Extraction method before Classification by using machine learning algorithms such as Genetic Algorithms, SVM and Clustering or by using statistical methods such as Particle Swarm Optimization, Principle Component Analysis, Gradual Feature Removal, and Mutual Information based Feature Selection.

ii) Integrating multiple classifiers and combining their result based on some criteria such as majority voting (Ensemble of classifiers).

iii) Integrating multiple classifiers where the output of one classifier is fed as an input to another classifier to refine the previous classification results (Stacking of Classifiers). It will also reduce the false alarms generated by a single classifier.

iv) Integrating feature selection/feature extraction approaches along with the multiple classifiers.

Observations based on analysis are as follows: (i) Single classifiers without feature selection are not giving good performance for attack detection.

(ii) A particular classifier is not providing good results for detecting all categories of attacks. Hence we can't say that a particular classifier is the best for attacks detection.

(iii) Data reduction technique is working well for refining the results for low-frequency attacks. It can be interpreted by looking the results of fuzzy rules for U2R attacks. The same may not work for all other attacks. It means many other factors affect the detection accuracy of the classifier for a particular attack such as the method of detection, sufficient features for learning the behavior of an attack, training dataset, data refinement and feature extraction.

B. Performance Analysis of Single Classifier Algorithms With limited Features

There is an improvement after feature selection with single classifiers as shown in Figure 11. For example, Decision tree [61] (C 4.5) is performing better with 12 features for DoS and Probe attacks detection. However, because of its incapability to detect low-frequency attacks, the author has experimented only with DoS and Probe attacks having a detection rate of 99.43% and 98.868% respectively. The 12 features are listed in Table VII in Section V. Hidden Naive Bayes [143] is providing similar results for DoS attacks with a detection rate of 99.6%. It relaxes the assumption of independence relation of variables and provides a significant improvement over Naive Bayes [121]. It achieves a detection rate of 99.6% over Naive Bayes with detection rate 96.65% for detection of DoS attacks. Author has not provided separate detection results for other attacks. So it is hard to say how it performs for other attacks. MMIFS with SVM [133] is providing average results for all four types of attacks. However, it provides better results for U2R and R2L attacks with a detection rate of 30.7% and 84.85%. The features employed by MMIFS technique is different for different attacks. It considers only 8 features for DoS, 12 features for Probe, 8 features for U2R and 10 features for R2L attacks. The features are listed in Table VII in Section V. The reasons for the degrading performance can be explained in terms of two major aspects.

(i) Low Detection Rate: Even after selecting features, a technique may not perform well for attack detection since the selected features are not good enough to learn the behavior of an attack. However, selecting and applying a suitable feature selection method is still an open challenge. The selection of an appropriate feature set is another aspect of improving the performance of the machine learning algorithms. For example,

MMIFS [133] is not considering feature P7 (land) which is very important for land (DoS) attacks detection. It also does not take features such as P10 (hot), P11 (num failed logins), P14 (root shell) and P16 (num root) into consideration which are very important for the detection of U2R and R2L attacks.

(ii) Algorithmic Drawback: Even after employing feature selection, the detection results are not good. It could be because of the algorithmic drawback of a classifier. The drawbacks of single classifiers with characteristics are mentioned in Section IV.

The limitations of single classifiers with feature selection can be **improved** by considering following factors:

(i) Each feature in the Feature Set should be relevant enough and nonsimilar to each other to learn the behavior of an attack. For example, P11 (num failed login), P14 (root shell) and P10 (hot) features are very important in the detection of U2R attacks and should be in the feature set for U2R attacks detection. There are various methods such as Gradual Feature Removal, Information gain, Chi-Square for feature selection. They may not provide the accurate features for all attacks. The researcher should not solely depend on the output of these methods.

(ii) Detection results and training time can be improved by integrating the classifiers so that another classifier can overcome the drawbacks of one classifier. For example, Clustering with SVM improves (CT SVM) [126] the training time and detection rate of SVM Classifier [120] as shown in Figure 10 and Figure 11. Clustering preprocesses the data and groups them into clusters based on the characteristics SVM Classifier is trained for one cluster. This improves the training time and accuracy of detection of SVM Classifier.

Observations based on analysis are as follows: (i) Employing feature selection does not necessarily mean that it will improve the detection results. The researcher should also focus on appropriate method employed for the feature selection. However, it will reduce the computational time of a classifier with less storage overhead. For example, MMIFS [133] with SVM with 8 features is providing detection rate of 78.69% for DoS attacks whereas SVM [120] without feature selection is achieving detection rate of 91.6% for DoS attacks.

(ii) If a feature set is working well for analyzing the behavior of a particular attack, it may not work well for other attacks. There is need to identify the behavior of each attack and accordingly to design a feature set for each attack.

(iii) If a Classifier is achieving 99.99% accuracy for DoS attacks or any other attacks. It should be the average of all types of DoS attacks such as Backdoor, Land, neptune, Pod, Smurf, teardrop etc. If a Classifier is achieving such a big accuracy 99.99% for one or two types of DoS attacks, it would be inappropriate to say that the Classifier is achieving 99.99% accuracy for that category of attack. For example, in Cluster Center and Nearest Neighbor (CANN) [144] approach one dimensional distance based feature is used to represent each data sample. The distance is the sum of two distances; first is distance between each data sample and its cluster center and second is distance between the data and its neighbor in the same cluster. Data samples are classified using k-Neighbor Classifier (k-NN). This approach comes in the category of single classifier with limited features in our categorization of machine learning algorithms. The approach claims to achieve 99.99% accuracy for DoS attacks and 99.98% accuracy for Probe attacks. They have utilized only 6 features namely P7(land), P9(urgent), P11(num failed logins), P18(num shells), P21(is host login), P20 (num outbound cmds).

Now as per our analysis of various research papers, out of these 6 features only feature7 (Land) is the most relevant feature for detecting Land attack. For this attack, CANN may achieve 99.99% accuracy. However, for other attacks such as such as teardrop, feature 8 (wrong fragment) is most relevant. To detect Back attack features (P5: src bytes, P6: dst bytes) [163] and features(P10: hot, P13: num compromised) [164] are most relevant features, calculated based on Rough Set theory and Information Gain measure respectively and hence should be included in the Feature Set. Olusola *et al.* [163] claims that feature P20 (outbound command count for FTP session) and the feature P21(hot login) are least relevant features for Intrusion Detection. The claim is provided based on the Dependency Ratio calculated for each feature based on a rough set theory which is 0.000 for both features (P20 ,P21) for all categories of DoS attacks. This justifies that CANN may work very good for Land attack but it has to improve its feature set for detecting other attacks.

C. Performance Analysis of Multiple Classifier Algorithms With All Features

We observed that multiple classifiers are performing better than single classifiers in term of detection rates as shown in Figure 10 and 12. If we consider the case of multiple classifiers without feature selection, Ensemble of ANN, SVM and MARS [124] is achieving the highest detection rate of 99.97% for DoS attacks. FC-ANN [132] is achieving a good detection rate 99.91% for detection of DoS attacks and 93.18% for detection of U2R attacks using KDD'99 dataset. Neuro-Fuzzy [147] is also achieving good detection rate for DoS (99.5%) using same dataset KDD'99. ANN with Elman network [130] experimented over DARPA 98 achieves the highest detection rate 100% for Probe attacks. FC-ANN [132] provides the highest detection rate 93.18% for U2R attacks over KDD'99 dataset. CSVAC [142] achieves good detection rate of 87% over KDD'99 dataset for R2L attacks Multiple NN [123] achieves 99.7% detection rate for specific U2R attack, i.e., Guess password attack. However, it has not been considered while comparison as it does not provide categorical results. The comparisons are performed with other techniques in this category. On an average, multiple classifiers with all features are performing good or average for high-frequency attacks (DoS and Probe), but they are resulting into average or poor detection rates for low-frequency attacks (U2R and R2L). However, they suffer from problems such as slow response time, high error rate, more storage requirement etc. For example Neuro Fuzzy [147] technique uses KDD'99 dataset and provides 99.5% detection rate for DoS, 84.1% detection rate for Probe, 41.1% detection rate for U2R and 31.5% detection rate for R2L attacks.

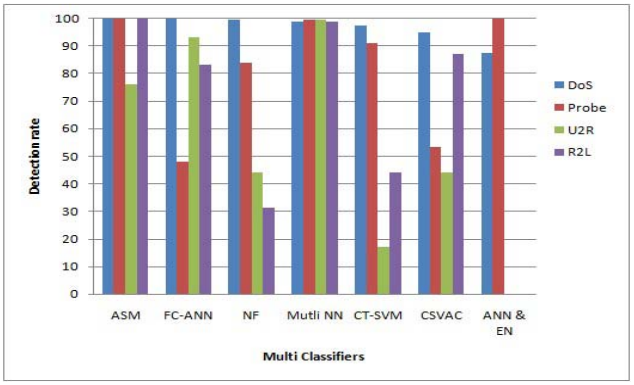

Fig. 12. Performance comparison of multiple classifiers with all features.

The limitation with this category of classifiers can be explained as.

(i) High Computational Cost: There are two reasons for having high computational cost:(a) Number of features are 41 which will enhance the computational cost of the classifier. It will also cause the problem of the curse of dimensionality which brings the sparsity in data as described in Section VII-A. It may lower the performance of classifier for the detection of U2R and R2L attacks since they are already limited in the KDD'99 training and test dataset. For DoS and Probe attacks detection most of the multiple classifiers are achieving 90-99% but the performance is not good for U2R and R2L attacks. (b) Multiple classifiers are slower than single classifiers in term of their training time because the data has to be processed by multiple classifiers to arrive at a common conclusion and sometimes the serial execution of classifiers where the output of one classifier becomes input to next classifier make them slower. For example, multiple classifier approach which combines Bayesian Clustering And Decision Tree (C4.5) [129] works in three stages. In the first stage, it classifies DoS, Probe and others (normal and U2R and R2L) using Decision Tree C4.5. In second stage, it separates normal connections from U2R and R2L using Bayesian Clustering and in third stage, it separates U2R from R2L again by using Decision Tree C4.5. This serial processing makes it much slower to process huge amount of data.

(ii) Complexity: Integrating many classifiers together makes the system complex. For example, In ANN with Elman Network [130], context nodes are additional nodes which are added to the neural network to keep the memory of past events. It will increase processing time of neural network and bring complexity into the network. Activation node of neural network receives the input from output of previously hidden nodes or input variable and context node (Elman network) which provides the data of past events. Moreover, the integration improves the detection rate and reduced retraining time but increases complexity and does not overcome the drawback of neural networks inability to detect low-frequency attacks such as U2R and R2L attacks as shown in Figure 12.

(iii) Average or Poor Detection Results for Low-Frequency Attacks: Some of the techniques are providing average performance whereas most of them are performing poor for the detection of Low-Frequency Attacks. The reasons are described in detail in Section VIII.

The performance of Multiple classifiers with all features can be **improved** by:

(i) Parallel Processing of the multiple classifiers Modules: Parallel processing of modules will reduce their computational time and increase their efficiency of attack detection. For example, in FC-ANN [132] approach, training data subsets are trained by different ANN classifiers which makes it very slow since training data subsets are processed one by one and then fuzzy aggregation module integrates the results. The training time of the classifier is 86625s (24h) which is quite high. Another example is multiple classifier approaches which uses the unsupervised technique (subspace clustering approach, DBSCAN and EA4RO algorithm) [26], partitions the feature space into small independent subspaces. DBSCAN is performed over each subspace to detect outliers one by one. It improves the detection rate but incurs in very high computational cost.

(ii) Integrating feature selection techniques with the multiple classifiers will increase their accuracy of attack detection and will reduce computational time.

Observations based on analysis are as follows:

(i) The detection rate of single classifiers is improved for most of the attacks by integrating it with other machine learning techniques. For example, SVM [120] is providing 91.6% detection rate for DoS attacks, 36.65% for Probe attacks, 12% for U2R attacks and 22% for R2L attacks. Whereas when it is integrated with Ant Colony networks (used for Clustering) named as CSVAC [142], it is achieving detection rate 94.84% for DoS attacks, 53.25% for Probe attacks, 44.23% for U2R attacks and 87% for R2L attacks. Both the techniques are experimented over KDD'99 dataset with 41 features. Hence multiple classifiers are performing better than single classifiers.

(ii) Most of the classifiers significantly improve the inefficiency of single classifiers in detecting U2R and R2L attacks. This is because those multiple classifiers perform the data filtering before sending the training data to the classifier. Clustering is one of the most popular techniques for sampling the data based on their behavior. It decreases the sparsity in data as similar data points are clustered in the same cluster. Each classifier is trained for the particular cluster to learn the behavior of the cluster data points. A cluster may refer to normal, DoS, Probe, U2R or R2L data records.

(iii) Integrating multiple classifiers may increase the computational time and complexity of a classifier making them more difficult to understand and design. (iv) Integrating a classifier with other classifier does not mean that it will improve the detection rate for all attacks. For example, ANN with all features [122] achieves 71.63% detection rate for Probe attacks detection when it is integrated with Fuzzy Clustering (FC-ANN) [132], its detection rate goes down to 48.12% for the same attack (Probe). Thus Fuzzy Clustering with ANN is not working well for Probe attacks detection. However, for other attacks, the detection rate is increasing. It can be interpreted by Figure 10 and Figure 12 for FC-ANN. The results have been obtained over KDD'99 dataset. It is because FC-ANN divides the training data into smaller subsets by using Fuzzy

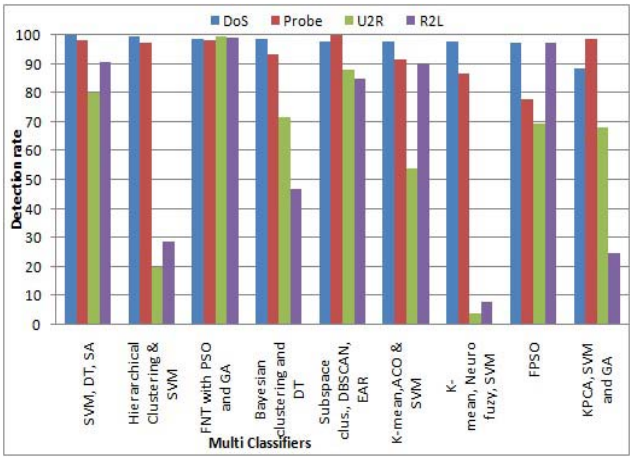

Fig. 13. Performance comparison of multiple classifiers with feature selection.

Clustering and then train the ANN Classifier for the individual subset. It may result in detection error for those attacks where training subset does not contain the sufficient number of attack connections.

D. Performance Analysis of Multiple Classifier Algorithms With limited Features

We observed that there is an improvement after feature selection with Multiple classifiers as shown in Figure 13. For example, Let us consider approach A (CSVAC) [142] which combines SVM with ant colony network and considers all 41 features over KDD'99. Another approach B [138] also combines SVM with ant colony network algorithm and uses 19 features obtained using GFS (Gradual Feature removal) over KDD'99. However, approach B also refines data using k-mean in the first stage. Approach B achieves the 97.67% detection rate for DoS attacks, 91.45% detection rate for Probe attacks. Also, 53.84% detection rate for U2R attacks and 90.34% detection rate for R2L attacks whereas approach A achieves the 94.84% detection rate for DoS attacks, 53.25% detection for Probe attacks, 44.23% detection rate for U2R attacks and 87% detection rate for R2L attacks. Approach B with feature selection and data refinement is performing better than approach A without feature selection. Most of the researchers have incorporated the feature selection strategy, i.e., filter based, wrapped based or hybrid based methods with multiple classifiers. They are performing good or average for detecting the high-frequency attacks (DoS and Probe) and low-frequency attacks (U2R and R2L). For example technique based on SVM, DT and SA with 23 features (listed in the Table VII) uses KDD'99 dataset and achieves 100% detection rate for DoS attacks, 98.35% detection rate for Probe attacks, 80% detection rate for U2R and 90.67% detection rate for R2L attacks. The limitation with this category of classifiers can be explained as.

(i) High computational time: However, feature selection has saved significant time of computation but it is still not much improved because of the execution of multiple modules of classifiers in serial order.

(ii) Complexity: These are very complex than single classifiers in design and working.

(iii) Average or good detection results for Low-Frequency Attacks: Few of the techniques are providing good performance whereas most of them are resulting in average performance. The detection results are not improved for all the techniques when they are integrated with other classifiers. Low-frequency attacks such as User to Root (U2R) and Remote to Local (R2L) preserves the similar characteristics as normal data. Moreover, there are various other reasons for having difficulty in detecting these attacks. The reasons are explained in details in Section VIII.

The performance of multiple classifiers with feature selection can be **improved** by:

(i) Parallel processing of multiple modules is necessary in order to provide the detection results on time. It is very important in case of real-time Intrusion Detection System where the notification of malicious activity has to be to notified as early as possible without causing much damage to the system. For example, various applications such as Online Banking, Online shopping and Cloud applications are very sensitive to malicious activity and a small malicious action can result in drastic results. This is explained with an example of a classifier in Section VII-C.

(ii)The selected features should be sufficient enough to analyze the behavior of an attack. Feature selection method provides the good results but these results should be further improved by researchers based on the signature of an attack. We have discussed the important features for each attack in Section III.

Observations based on analysis are as follows: (i) Data refinement and dividing the data into clusters improves the accuracy of detection. The discussed approach B in [138] performs extensive data refinement using k-mean and ACO before passing the training data to SVM. Approach A performs the data sampling. It selects the random data points in the first iteration and creates support vectors for selected data points. In the next iterations, only those data points are selected which are closer to the generated support vectors and classifier is retrained for new data points.

(ii) If a technique is achieving highest detection rate for a particular attack, It may not achieve the same performance for detecting the other attacks. For example, Hierarchical Clustering with SVM technique [134] is considering 19 features and using KDD'99 dataset for detecting all attacks (DoS, Probe, U2R and R2L) and is achieving a detection rate of 99.5% for DoS, 97.5% for Probe, 19.7% for U2R and 28.8% for R2L. It means we should not use the same technique to detect all attacks, i.e., if a technique is good in DoS and Probe, it may not be good in U2R and R2L attacks.

(iii) If an optimal subset of features is good for detecting one attack, it may not be good for detecting other attacks. Most of the authors are considering a same optimal subset of features for detecting all four categories of attacks and their technique is providing major variation in terms of detection rate for different attacks. Since the behavior of an attack is different than other attacks, there is need to train the classifier for different categories of attacks using

the different optimal subset chosen for different attack. For example FPSO [135] is using same 16 features subset {P1,P5,P6,P8,P9,P10,P11,P13,P16,P17,P18,P19,P23, P24,P32,P33}for detecting DoS, Probe, U2R and R2L. However, it results in good detection results for DoS (97.22%) and R2L(97.22%) and average results for Probe(77.77%) and U2R(69.44%). Here features P3 (service), P7 (land) and P14 (root shell) are not considered which are most important features for attack detection.

(iv) Multiple classifier with selected features are performing better than other categories of classifiers.

An exhaustive literature study of intrusion detection techniques with respect to each category of classification techniques is carried out and critically analyzed. Various inferences are drawn by comparing results reported by researchers. The observations are analyzed thoroughly. The limitations associated with each category of the technique is discussed and viable solutions to overcome the limitations are provided. At the end, conclusions drawn with respect to each classification are mentioned to provide scope for further improvement.

VIII. Issues in Detecting Low-Frequency Attacks

Machine learning algorithms work on the statistics of the data obtained from attack data set. DoS and Probe attacks can be detected easily by careful examination of the statistics of the connections at the vulnerable host machine whereas it is hard to detect the low-frequency attacks such as U2R and R2L even by careful examination of the statistics of the connections using KDD'99 dataset. This is because of the following reasons:

(i) The connection statistics of low-frequency attacks are very similar to the normal connection.

(ii) There exist similarity in the behavior of U2R and R2L connection records. Hence, it is also difficult to differentiate U2R and R2L attacks itself from each other. In fact, a U2R attack is one of the variations of R2L attack. In R2L attack, a user does not have the local access to the machine. To access the root privileges, he has to first access a normal user's account by using various account hijacking exploits. Then after login as a normal user, he can launch further exploits to gain root privileges whereas, in a U2R attack, the attacker has unprivileged local access to the victim machine.

(iii) Low-frequency attacks can be launched in a single connection. Information provided by the KDD'99 dataset about the connection is not sufficient. Although some of the Content features are present in the KDD'99 dataset (refer Table II in Section III) such as num failed logins (P11), root shell (P14), num compromised (P13), root shell(14), etc. but they are not enough for attack identification. For example, loadmodule attack (U2R) loads two dynamically loadable kernel drivers of the currently running system and creates special devices in the /dev directory to use those modules. Because of a bug in the way loadmodule sanitizes environment, an unauthorized user can gain root access on the local machine. The attack can be detected by the keyword spotting in the user's session to find strings 'set $IFS='V' and 'loadmodule' [53]. This kind of keyword spotting is difficult to achieve with machine learning algorithms using KDD'99 dataset.

(iv) The number of U2R and R2L samples, present in the training and testing dataset of KDD'99 are very less when compared to the DoS and Probe attacks. The insufficient learning of such attacks makes the classifier less suitable to detect such attacks. Moreover, imbalanced distribution of data, make the classifier to treat such attacks as normal attacks.

(v) Activities performed by these attacks may be similar in terms of a number of file creation, root shell login, sum of operations performed as root, etc. In such a case identifying the low-frequency attacks become more difficult. However, careful examination of the system call traces for the presence of specific modules or processes, the suspicious sequence of system calls, invocation of specific commands, etc. may provide some clues to identify the attack activity in the system. For example, Ffbconfig attack exploits a buffer overflow (U2R). It configures the Creator Fast Frame Buffer (FFB) Graphics Accelerator which is a part of FFB Configuration software package, SUNWffbcf. The attack can be detected by examining the system call traces of the system for the presence of '/usr/sbin/ffbconfig/' command with an oversized argument for '-dev' parameter [53].

(vi) A very high accuracy (around 90-99%) is achieved in some approaches in detecting those attacks but we can't say that these techniques will achieve the same accuracy for detecting the unseen attacks or newly generated U2R or R2L attacks. Since the techniques are validated over the test database of KDD'99 which contains the features values of the attacks that may be sufficient to separate them from DoS and Probe. For example, a dictionary attack is an R2L attack in which attacker makes repeated guesses of username and passwords to gain access to some machine remotely. The attack can be detected by examining two features: session protocol of every service (P2) and num of failed login attempts (P11) over a period. But, if the feature values are not providing sufficient information which may happen if victim password is not strong enough and attacker accesses the victim machine in one or two guesses such as by entering his phone no or school name etc. The feature values for this attack will be similar to a normal connection. In this case, machine learning algorithm will not work efficiently to detect these attacks.

It is difficult to detect low-frequency attacks just by examination of network features. The issues in detecting low-frequency attacks have been identified and discussed. The possible viable solutions to detect such attacks such as buffer overflow, password cracking, dictionary attack, virus, etc. have been discussed. One can refer our recent work [76], [165] for detecting these attacks using system call analysis. In our another recent work [77], we have considered both system call and network features for detection of low-frequency attacks.

IX. Data Mining Tools for Machine Learning

There exist many tools that support the implementation of various machine learning methodologies. Some of them are described below.

A. Weka

Waikato Environment for Knowledge Analysis (Weka) [166] is a machine learning software tool, developed by University of Waikato, New Zealand in 1993. Although many changes have been incorporated till now. It is an open source tool which is freely available under GNU General Public License and written in JAVA. Weka supports various data analysis tasks such as data pre-processing, feature selection, classification, clustering, regression and visualization. The tool takes input as a set of records in the flat file (.ARFF files) where a set of attributes describes each record. It is easy to use due to Graphic User Interfaces (GUIs) provided by the tool.

B. Scikit-Learn

Scikit-learn is an open source machine learning the library, developed as Google summer of code project by David Cournapeau in 2007 [167]. It is written in python and incorporates python numerical and scientific libraries like NumPy and SciPy and other libraries like Panda, matplotlib, etc. It provides various efficient tools for machine learning like classification, clustering, regression algorithms. It also supports methods for feature extraction and provides learning tutorials to understand the concepts.

C. TensorFlow

TensorFlow [168] is an open source software library for machine learning applications, developed by Google Brain Team. It was released under Apache 2.0 open source license on Nov 2015. Version 1.0.0 is recently released in Feb 2017. TensorFlow is a very useful tool for deep learning as it provides support for building and training neural networks. Data flow graphs are used to create machine learning models and perform computations. Data arrays are edges between nodes of graphs, called as tensors. TensorFlow supports multiple APIs such as python, C++, Go, Java, Haskell and Rust APIs. Third party packages are available for Julia, Scala and R. It can run multiple CPUs and GPUs and supports different 64 bits OS such as Linux, Windows, MacOS, Android and iOS, etc. TensorFlow Lite is a recent release for Android.

D. KMINE

Konstanz Information Miner (KNIME [169] is an open source tool for data analytics, developed at University of Konstanz, released under a dual licensing scheme. It uses the data pipelining concepts to integrate the components of machine learning and data mining. It provides GUI to perform various tasks such as data loading, transformation, feature extraction, modeling and visualization. It is written in Java but provides a wrapper to run other codes like python, perl. It provides the processing of large data volumes. For example, it can analyze 300 million customer addresses, 10 million molecular structure and 20 million cell images. It integrates open source projects such as ML algorithms from Weka, R packages, LibSVM, JFreeChart, ImageJ, etc. using plugins.

E. RapidMiner

RapidMiner [170] is a software developed by RapidMiner company for data mining applications. It provides an integrated environment for data pre-processing, machine learning, deep learning, text mining etc. It provides both commercial and free edition. The free edition is named as RapidMiner Studio which is limited to 1 logical processor and 10,000 rows and can be obtained under AGPL license. It uses client/server model where a server is basically hosted on the cloud platform. It provides a template based framework and removes the need for coding. It supports text mining, image mining, video mining, and social network analysis. The import formats supported by it are SQL, TXT, XML, XLS, etc. It performs data extraction, transformation, analysis and visualization.

F. Environment for Developing KDD-Applications Supported by Index-Structure (ELKI)

ELKI [171] is an open source software for data mining applications, developed at the Ludwig Maximilian University of Munich, Germany. It emphasizes on unsupervised machine learning methods such as K-mean clustering, K-medians clustering, DBSCAN, OPTICS, Expectation-maximization, Hierarchical clustering Canopy clustering, etc. It also provides data index structures like R-ree, R*-tree, M-tree and K-d tree. The advanced mining algorithms and their interaction with database index structure is evaluated using many parameters like ROC, histogram, Scatterplot, etc.

G. Massive Online Analysis (MOA)

MOA [172] is a popular open source data stream mining tool to perform big data streaming in real time. It consists of various machine learning algorithms for classification, clustering, regression and outlier detection, etc. It is written in Java and can be extended for newer algorithms, streams and evaluation methods. It provides storable settings for real and synthetic data stream for conducting repeatable experiments.

We discussed some of the important data mining tools used for data analysis using machine learning algorithms. Some of them also support deep learning algorithms. Most of these tools provide easy to use GUI interface that can be easily used by researchers in their research domain. Some other machine learning libraries are Apache SAMOA [173] and MLlib (Spark) [174], etc. those can also be used by researchers.

X. Future Directions

Deep learning is an advancement of the neural network. Deep learning uses the subsequent layers of information-processing in some hierarchy for classification or feature representation. It makes use of the deep networks having multiple layers of processing. It consists of input tier providing the basic data and followed by consecutive hidden layers which analyze data and output is produced. It has gained popularity in recent years. The existing IDS can be improved by embracing this latest technique. Deng and Yu [175] provided the categorization of deep learning methods based on their

TABLE XII
RECENT WORKS BASED ON APPLICATION OF DEEP LEARNING FOR INTRUSION DETECTION

References	Feature Extraction	Algorithm
[177]	Manually normalized	DNN with Bayesian Calibration
[178]	Stacked Auto Encoder	Stacked Auto Encoder/ANN
[179]	Auto Encoder	Deep Belief Network
[180]	Stacked Auto Encoder	Extreme Learning Machine
[181]	Manually normalized	Deep Belief Network
[182]	Manually normalized	DNN with Recurrent Neural Network
[183]	Monte Carlo Tree Search	Convolution Neural Network
[184]	Manually normalized	Restricted Boltzman Machine
[185]	Manually normalized	Deep Belief Network
[186]	Stacked Auto Encoder	Logistic Regression
[187]	Deep Belief Network	Support Vector Machine
[188]	Manually normalized	Convolution Neural Network

architecture into following types: generative (unsupervised), discriminative (supervised) and hybrid. Unsupervised deep learning or generative architectures make use of following methods: Auto Encoder (AE) and Boltzman Machine (BM). Similar to ANN, AE makes use of hidden layers; however, it has only three hidden layers. The nodes in the input layer and output layer are same. The hidden nodes are used to reduce the feature dimensionality and provide the new feature set [176]. A different set of features are learned in cascade depths to train the more precisely. BM takes the stochastic decision using the neuron's structure of binary units. Deep BM (DBM) has a cascaded structure whereas Restricted BM has no connections among the hidden units. The multiple layers which are stacked one by one form a deep belief networks (DBN). Supervised learning is used to distinguish some parts of data and has been used for pattern classifications. Convolution Neural Network (CNN) is an example of supervised learning which provides fast learning. CNN uses three fields: local receptive fields, shared weights and pooling. Hybrid approach makes use of both the methods. An example of hybrid architecture is Deep Neural Networks (DNN). DNN provides a fully connected hidden layer forming cascaded multilayer networks.

The use of deep learning for image classification is quite popular. However, the challenge lies in adopting the deep learning for attack detection in network traffic. In recent years, it has been applied deep learning for intrusion detection as shown in Table XII. Seok and Kim [188] have employed deep learning for attack detection which is based on the conversion of malware code into the image and applying CNN which takes these images as input to learn attack features. Also, most of the work for deep learning based IDS uses this approach for reducing the dimensionality of features. It has many advantages. Combining the supervised and unsupervised approaches of deep learning improve the detection results of traditional approaches [189]. It helps in developing new methods on network security which are more certain than traditional machine learning approaches. Deep learning is adaptable to the changing context of data as it performs the exhaustive data analysis. However, the use of deep learning for attack analysis is still challenging and open area for researchers to work on. The resources required for training the network are also quite huge. Deep learning is suitable to be applied when it is difficult to find the correlation between raw input and target class. A reliable intrusion detection system should be able to handle the noisy inputs and large discrete or continuous data.

Reinforcement learning (RL) is another interesting area of research. Reinforcement learning (RL) is one of the machine learning algorithms where multiple agents and machine work/interact together to learn the behavior within a particular context and improve the performance for attack detection. Sensors or agents sense the environment in discrete time intervals and the input is mapped to locate the state information. Once RL agents execute the action and feedback is observed from the environment. Correct actions of agents are rewarded by the environment, called reinforcement signal. Agents then leverage the rewards and improve the knowledge about the environment to select the next action. Some of the researchers have applied RL to detect the distributed denial of service attacks [190].

Servin and Kudenko [191] proposed a hierarchical distributed architecture for intrusion detection in which multiple network-agents learns to capture the local state information. All the sensors communicate to the agent up in the hierarchy. The topmost agent decides when to fire and generate an alarm. Each agent uses the slightly modified version of the Q-learning and simple exploration/exploitation strategy to learn the actions and execute in a particular state. Random selection is carried out to choose between normal and abnormal states and global state of the network is simulated. The approach achieves an accuracy of 98.9% with 1.1% error rate with two sensor agents in a self-generated dataset.

In order to improve the efficiency and improve the computational power, Xu and Xie [192] applies the RL for host based intrusion detection. They have applied Markov reward process model for modeling the behavior of the host. RL prediction method makes use of the temporal difference learning algorithm [193] for learning the behavior of the processes. It helps in detecting the abnormal process behavior of the applications running on the host. The use of RL for predicting the behavior of normal system call sequences has helped in improving the accuracy. They have obtained 100% detection rate and 20 sec of training time using MIT lpr dataset [194]. The computational cost is very low in comparison to traditional ML algos such as HMM, RIPPER etc. The summary of some of the crucial research challenges associated with deep learning and reinforcement learning approaches are discussed as follows:

- One of the primary challenge using deep algorithms is to generate/obtain a lot of data for training & classification algorithm. For example, in order to make an IDS learn the various attack scenarios, researchers supply terabytes of data to the deep learning classifier to train itself. Availability of sufficient data specific to the problem domain is one of the crucial challenges.
- It will be challenging to adopt deep learning algorithm for real-time classification because of the level of complexity involved in training huge amount of data. Most of the existing work apply deep learning for feature extraction and dimensionality reduction only [176].
- Most of the existing deep learning methods are suitable for pattern and image recognition. However, how to apply deep learning to classify the network traffic and/or system

logs properly, is still a challenge. Some of the deep learning algorithm's like Convolution Neural Network (CNN) and Deep Belief Network (DBN) has proven to be good classifiers. However, the experimental work is in progress to determine the efficiency and reliability of these learning algorithms to detect attacks [195].

- Another challenge in the deep learning algorithms is the requirement of high-performance hardware to process the huge training data. Machines need to have sufficient processing power to solve the real world problems. To reduce the training time and improve efficiency, researchers will require multi-core high performing GPUs which are very costly and consume more power. For example, Monte Carlo Tree search integrated with deep neural network requires 48 CPUs, 8 GPUs for conducting 40 multi-threaded search [183].
- The growth in computer memory and computational power is possible through parallel and distributed computing. Researchers can work in this direction to cope with the issues related to communication and computation management to scale the deep learning algorithms for huge datasets.
- Reinforcement learning (RL) is one of the growing field and research in this direction towards attacks detection is still going on. The slow study speed problem of RL affects the feasibility of multi-agent study in the real world. How to speed up the performance of multi-agent classifier is another important research challenge [196].
- In the Multi-agent RL system, there has to be proper coordination among each of the RL agent members. Therefore, designing a fast and effective way for communication is another important research concern. Researchers are still working on How to apply RL for filtering the network traffic more accurately.

Deep Learning and Reinforcement Learning are the future research directions in the field of intrusion detection for researchers. Deep Reinforcement Learning [197] is the next step in this direction to make the learning effective for huge collection of data. It has been applied for pattern classification and resource control purpose in past years. Researchers can apply it for intrusion detection applications.

XI. Conclusion

The increasing rate of intrusions in the network and host machines have badly affected the security and privacy of users. Researchers have extensively worked on various solutions to detect intrusions. The security aspects of intrusion detection using machine learning approach have been considered in our paper. We have described various types of attacks in the network and host systems with the brief description of their attack features. The analysis performed, reveals that if a technique is performing well for detecting an attack, it may not perform same for detecting other attacks. Hence, the relevance of a technique for specific attacks has been presented by classifying various machine learning techniques for each type of attack.

The critical performance analysis of various machine learning algorithms has been done in an evolutionary way. The comparison has been carried out with single classifier approaches and multiple classifier approaches. The influence of a classifier with other classifier is not only analyzed but also the influence of a feature subset with the classifier is analyzed. We have shown that even if an optimal feature set is sufficient for analyzing the behavior of an attack, it is not good for analyzing the behavior of other attacks. Hence, there is a need to define the optimal feature subset and a suitable technique for each type of attack as the behavior of an attack varies from each other.

The difficulties associated with detecting the low-frequency attacks using machine learning techniques over network dataset have been described. It motivates researchers to work on other solutions to detect the Low-frequency attacks. Future research directions are provided to help researchers exploring more efficient solutions for attack detection.

Existing literature is described which are based on similar techniques with most of the popular datasets as on date to generalize our observations. All the techniques have not been implemented to evaluate the performance to ensure that results are reproducible. This remains a limitation of our paper and we are very keen to improve this as a future work. In future, we would also like to propose an attack detection model especially for improving the performance of low-frequency attacks by exploring deep learning approaches. Later, various issues will be focused with IDS techniques when these are applied to dynamic and changing network environment such as Cloud Computing etc.

References

[1] BBC News. (2008). *Estonia Fines Man for 'Cyber War'*. [Online]. Available: http://news.bbc.co.uk/2/hi/technology/7208511.stm

[2] L. Dignan. (2008). *Amazon Exploits Its S3 Outage*. [Online]. Available: http://www.zdnet.com/article/amazon-explains-its-s3-outage/

[3] M. Dekker, D. Liveri, and M. Lakka, "Cloud security incident reporting: Framework for reporting about major cloud security incidents," ENSIA, St. Paul, MN, USA, Rep. TP-04-13-105-EN-N, 2013.

[4] DDoS. (2014). *Ello Social Network Hit by Suspected Bloody DDoS Attack*. [Online]. Available: http://www.ddosattacks.net/ello-social-network-hit-by-suspected-bloody-dDoS-attack/

[5] S. Panjwani, S. Tan, K. M. Jarrin, and M. Cukier, "An experimental evaluation to determine if port scans are precursors to an attack," in *Proc. IEEE Int. Conf. Depend. Syst. Netw. DSN*, 2005, pp. 602–611.

[6] CISCO. (2014). *Cisco Anual Report*. [Online]. Available: www.cisco.com/web/offer/gist-ty2-asset/Cisco-2014-ASR.pdf

[7] "Cisco annual cyber security report," CISCO, San Jose, CA, USA, Rep., 2017.

[8] SNORT. (2017). *Snort 2.9.7.6*. [Online]. Available: https://www.snort.org/

[9] OISF. (2018). *Suricata 4.0.4*. [Online]. Available: https://suricata-ids.org/about/

[10] T. F. Lunt, "Ides: An intelligent system for detecting intruders," in *Proc. Symp. Comput. Security Threat Countermeasures*, 1990, pp. 30–45.

[11] L. Ertöz *et al.*, "MINDS-minnesota intrusion detection system," in *Next Generation Data Mining*. Cambridge, MA, USA: MIT Press, 2004, pp. 199–218.

[12] KDD. (1999). *KDD Cup 1999 Data*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[13] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Canberra, ACT, Australia, 2015, pp. 1–6.

[14] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and survey of collaborative intrusion detection," *ACM Comput. Surveys*, vol. 47, no. 4, p. 55, 2015.

[15] A. Torkaman, G. Javadzadeh, and M. Bahrololum, "A hybrid intelligent HIDS model using two-layer genetic algorithm and neural network," in *Proc. IEEE 5th Conf. Inf. Knowl. Technol. (IKT)*, 2013, pp. 92–96.

[16] R. Puzis, M. D. Klippel, Y. Elovici, and S. Dolev, "Optimization of NIDS placement for protection of intercommunicating critical infrastructures," in *Proc. IEEE Int. Conf. Intell. Security Informat.*, Taipei, Taiwan, 2008, pp. 191–203.

[17] A.-S. K. Pathan, *The State of the Art in Intrusion Prevention and Detection*. Boca Raton, FL, USA: CRC Press, 2014.

[18] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Washington, DC, USA, 1989, pp. 593–605.

[19] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann, 1993.

[20] C. Cortes and V. Vapnik, "Support vector machine," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[21] S. Kumar, *Survey of Current Network Intrusion Detection Techniques*, Washington Univ., St. Louis, MO, USA, pp. 1–18, 2007.

[22] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, Jan. 2016.

[23] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Security*, vol. 28, nos. 1–2, pp. 18–28, 2009.

[24] P. Kumar *et al.*, "A novel approach for security in cloud computing using hidden Markov model and clustering," in *Proc. IEEE World Congr. Inf. Commun. Technol. (WICT)*, 2011, pp. 810–815.

[25] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[26] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised network intrusion detection systems: Detecting the unknown without knowledge," *Comput. Commun.*, vol. 35, no. 7, pp. 772–783, 2012.

[27] J. Yang, T. Deng, and R. Sui, "An adaptive weighted one-class SVM for robust outlier detection," in *Proc. Chin. Intell. Syst. Conf.*, 2016, pp. 475–484.

[28] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Exp. Syst. Appl.*, vol. 41, no. 4, pp. 1690–1700, 2014.

[29] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 184–208, 1st Quart., 2015.

[30] Y. Zhang, W. Lee, and Y.-A. Huang, "Intrusion detection techniques for mobile wireless networks," *Wireless Netw.*, vol. 9, no. 5, pp. 545–556, 2003.

[31] C. Kolias, V. Kolias, and G. Kambourakis, "TermID: A distributed swarm intelligence-based approach for wireless intrusion detection," *Int. J. Inf. Security*, vol. 16, no. 4, pp. 401–416, 2016.

[32] M. Halilovic and A. Subasi, "Intrusion detection on smartphones," *arXiv e-print 1211.6610*, pp. 1–8, Nov. 2012.

[33] A. Karim, R. Salleh, and M. K. Khan, "SMARTbot: A behavioral analysis framework augmented with machine learning to identify mobile botnet applications," *PLoS ONE*, vol. 11, no. 3, pp. 1–35, 2016.

[34] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "'Andromaly': A behavioral malware detection framework for android devices," *J. Intell. Inf. Syst.*, vol. 38, no. 1, pp. 161–190, 2012.

[35] A. K. Sikder, H. Aksu, and A. S. Uluagac, "6thSense: A context-aware sensor-based attack detector for smart devices," in *Proc. 26th USENIX Security Symp.*, 2017, pp. 397–414.

[36] P. Faruki *et al.*, "Android security: A survey of issues, malware penetration, and defenses," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 998–1022, 2nd Quart., 2015.

[37] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "Intrusion detection techniques in cloud environment: A survey," *J. Netw. Comput. Appl.*, vol. 77, pp. 18–47, Jan. 2017.

[38] S. Anwar *et al.*, "Cross-VM cache-based side channel attacks and proposed prevention mechanisms: A survey," *J. Netw. Comput. Appl.*, vol. 93, pp. 259–279, Sep. 2017.

[39] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Comput. Sci.*, vol. 60, pp. 708–713, Dec. 2015.

[40] N. F. Haq *et al.*, "Application of machine learning approaches in intrusion detection system: A survey," *Int. J. Adv. Res. Artif. Intell.*, vol. 4, no. 3, pp. 9–18, 2015.

[41] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2015.

[42] D. Csubak and A. Kiss, "OpenStack firewall as a service rule analyser," in *Proc. Int. Conf. Human Aspects Inf. Security Privacy Trust*, 2016, pp. 212–220.

[43] P. S. Kenkre, A. Pai, and L. Colaco, "Real time intrusion detection and prevention system," in *Proc.3rd Int. Conf. Front. Intell. Comput. Theory Appl. (FICTA)*, 2015, pp. 405–411.

[44] P. Deshpande, A. Aggarwal, S. Sharma, P. S. Kumar, and A. Abraham, "Distributed port-scan attack in cloud environment," in *Proc. 5th Int. Conf. Comput. Aspects Soc. Netw. (CASoN)*, Fargo, ND, USA, 2013, pp. 27–31.

[45] M. J. Schoelles and W. D. Gray, "Argus: A suite of tools for research in complex cognition," *Behav. Res. Methods Instrum. Comput.*, vol. 33, no. 2, pp. 130–140, 2001.

[46] A. Crenshaw. (2008). *OSfuscate: Change Your Windows OS TCP/IP Fingerprint to Confuse P0f, NetworkMiner, Ettercap, Nmap and Other OS Detection Tools*. [Online]. Available: http://www.irongeek.com/security/osfuscate-change-your-windows-os-tcp-ip-fingerprint-to-confuse-p0f-networkminer-ettercap-nmap-and-other-os-detection-tools.htm

[47] D. Norton, *An Ettercap Primer*. Singapore: SANS Inst. InfoSec Reading Room, 2004.

[48] N. Hoque, M. H. Bhuyan, R. C. Baishya, D. Bhattacharyya, and J. K. Kalita, "Network attacks: Taxonomy, tools and systems," *J. Netw. Comput. Appl.*, vol. 40, pp. 307–324, Apr. 2014.

[49] G. Mantas, N. Stakhanova, H. Gonzalez, H. H. Jazi, and A. A. Ghorbani, "Application-layer denial of service attacks: Taxonomy and survey," *Int. J. Inf. Comput. Security*, vol. 7, nos. 2–4, pp. 216–239, 2015.

[50] F. Iglesias and T. Zseby, "Analysis of network traffic features for anomaly detection," *Mach. Learn.*, vol. 101, nos. 1–3, pp. 59–84, 2014.

[51] E. Guillén, J. Rodríguez, R. Paez, and A. Rodriguez, "Detection of non-content based attacks using GA with extended KDD features," in *Proc. World Congr. Eng. Comput. Sci.*, 2012, pp. 30–35.

[52] D. Kumar, "DDoS attacks and their types," in *Network Security Attacks and Countermeasures*. Hershey, PA, USA: Inf. Sci. Ref., 2016, p. 197.

[53] MIT. (1999). *Darpa Intrusion Detection Attacks Database*. [Online]. Available: http://www.ll.mit.edu/ideval/docs/attackDB.html

[54] M. Malekzadeh, M. Ashrostaghi, and M. S. Abadi, "Amplification-based attack models for discontinuance of conventional network transmissions," *Int. J. Inf. Eng. Electron. Bus.*, vol. 7, no. 6, p. 15, 2015.

[55] S. Maiti, C. Garai, and R. Dasgupta, "A detection mechanism of DoS attack using adaptive NSA algorithm in cloud environment," in *Proc. IEEE Int. Conf. Comput. Commun. Security (ICCCS)*, 2015, pp. 1–7.

[56] T. Bass, A. Freyre, D. Gruber, and G. Watt, "E-mail bombs and countermeasures: Cyber attacks on availability and brand integrity," *IEEE Netw.*, vol. 12, no. 2, pp. 10–17, Mar./Apr. 1998.

[57] T. Halagan, T. Kováčik, P. Trúchly, and A. Binder, "Syn flood attack detection and type distinguishing mechanism based on counting bloom filter," in *Proc. Inf. Commun. Technol. EurAsia Conf.*, 2015, pp. 30–39.

[58] E. Bou-Harb, M. Debbabi, and C. Assi, "Cyber scanning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1496–1519, 3rd Quart., 2014.

[59] A. K. Kaushik, E. S. Pilli, and R. C. Joshi, "Network forensic system for port scanning attack," in *Proc. IEEE 2nd Int. Adv. Comput. Conf. (IACC)*, 2010, pp. 310–315.

[60] L. Aniello, G. Lodi, and R. Baldoni, "Inter-domain stealthy port scan detection through complex event processing," in *Proc. 13th Eur. Workshop Depend. Comput.*, 2011, pp. 67–72.

[61] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Comput. Commun.*, vol. 34, no. 18, pp. 2227–2235, 2011.

[62] D. Mankins, R. Krishnan, C. Boyd, J. Zao, and M. Frentz, "Mitigating distributed denial of service attacks with dynamic resource pricing," in *Proc. 17th Annu. Comput. Security Appl. Conf. (ACSAC)*, 2001, pp. 411–421.

[63] G. Helmer *et al.*, "A software fault tree approach to requirements analysis of an intrusion detection system," *Requirements Eng.*, vol. 7, no. 4, pp. 207–220, 2002.

[64] A. Sridharan, T. Ye, and S. Bhattacharyya, "Connectionless port scan detection on the backbone," in *Proc. 25th IEEE Int. Perform. Comput. Commun. Conf. (IPCCC)*, Phoenix, AZ, USA, 2006, p. 10.

[65] E. Raftopoulos, E. Glatz, X. Dimitropoulos, and A. Dainotti, "How dangerous is Internet scanning?" in *Proc. Int. Workshop Traffic Monitor. Anal.*, 2015, pp. 158–172.

[66] M. Rostamipour and B. Sadeghiyan, "Network attack origin forensics with fuzzy logic," in *Proc. IEEE 5th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, 2015, pp. 67–72.

[67] S. Bahl and S. K. Sharma, "A minimal subset of features using correlation feature selection model for intrusion detection system," in *Proc. 2nd Int. Conf. Comput. Commun. Technol.*, 2016, pp. 337–346.

[68] C. Edge, W. Barker, B. Hunter, and G. Sullivan, "Malware security: Combating viruses, worms, and root kits," in *Enterprise Mac Security*, Berkeley, CA, USA: Apress, 2016, pp. 221–242.

[69] D. G. Johnson and T. M. Powers, "Computer systems and responsibility: A normative look at technological complexity," *Ethics Inf. Technol.*, vol. 7, no. 2, pp. 99–107, 2005.

[70] A. A. Ghorbani, W. Lu, and M. Tavallaee, "Network attacks," in *Network Intrusion Detection and Prevention*. Cham, Switzerland: Springer Int., 2010, pp. 1–25.

[71] P. K. Manadhata and J. M. Wing, "An attack surface metric," *IEEE Trans. Softw. Eng.*, vol. 37, no. 3, pp. 371–386, May/Jun. 2011.

[72] S. Singh and S. Silakari, "A survey of cyber attack detection systems," *Int. J. Comput. Sci. Netw. Security*, vol. 9, no. 5, pp. 1–10, 2009.

[73] M. K. Sabhnani and G. Serpen, "KDD feature set complaint heuristic rules for R2L attack detection," in *Proc. Security Manag.*, 2003, pp. 310–316.

[74] K. S. Wutyi and M. M. S. Thwin, "Heuristic rules for attack detection charged by NSL KDD dataset," in *Genetic and Evolutionary Computing*. Cham, Switzerland: Springer Int., 2016, pp. 137–153.

[75] P. Mishra, E. S. Pilli, and R. C. Joshi, "Forensic analysis of e-mail date and time spoofing," in *Proc. IEEE 3rd Int. Conf. Comput. Commun. Technol. (ICCCT)*, 2012, pp. 309–314.

[76] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "VAED: VMI-assisted evasion detection approach for infrastructure as a service cloud," *Concurrency Comput. Pract. Exp.*, vol. 29, no. 12, pp. 1–21, 2017.

[77] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "PSI-NetVisor: Program semantic aware intrusion detection at network and hypervisor layer in cloud," *J. Intell. Fuzzy Syst.*, vol. 32, no. 4, pp. 2909–2921, 2017.

[78] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Security*, vol. 31, no. 3, pp. 357–374, 2012.

[79] V. H. Garcia, R. Monroy, and M. Quintana, "Web attack detection using ID3," in *Professional Practice in Artificial Intelligence*. Cham, Switzerland: Springer Int., 2006, pp. 323–332.

[80] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann, 1993.

[81] A. L. Prodromidis and S. J. Stolfo, "Cost complexity-based pruning of ensemble classifiers," *Knowl. Inf. Syst.*, vol. 3, no. 4, pp. 449–469, 2001.

[82] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, 1997.

[83] M. F. Augusteijn and B. A. Folkert, "Neural network classification and novelty detection," *Int. J. Remote Sens.*, vol. 23, no. 14, pp. 2891–2902, 2002.

[84] M. M. Moya, M. W. Koch, and L. D. Hostetler, "One-class classifier networks for target recognition applications," Sandia Nat. Labs., Albuquerque, NM, USA, Rep. SAND-93-0084C, 1993.

[85] S. Albrecht, J. Busch, M. Kloppenburg, F. Metze, and P. Tavan, "Generalized radial basis function networks for classification and novelty detection: Self-organization of optimal Bayesian decision," *Neural Netw.*, vol. 13, no. 10, pp. 1075–1093, 2000.

[86] A. Jagota, "Novelty detection on a very large number of memories stored in a hopfield-style network," in *Proc. IEEE Seattle Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 2. Seattle, WA, USA, 1991, p. 905.

[87] D. Martinez, "Neural tree density estimation for novelty detection," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 330–338, Mar. 1998.

[88] A. Bivens *et al.*, "Network-based intrusion detection using neural networks," *Intell. Eng. Syst. Artif. Neural Netw.*, vol. 12, no. 1, pp. 579–584, 2002.

[89] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell.*, Montreal, QC, Canada, 1995, pp. 338–345.

[90] A. McCallum and K. Nigam, "A comparison of event models for naive Bayes text classification," in *Proc. AAAI Workshop Learn. Text Categorization*, vol. 752. Madison, WI, USA, 1998, pp. 41–48.

[91] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive Bayes for text categorization revisited," in *Proc. Aust. Conf. Artif. Intell.*, Cairns, QLD, Australia, 2004, pp. 488–499.

[92] L. Jiang, H. Zhang, and Z. Cai, "A novel Bayes model: Hidden naive Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 10, pp. 1361–1371, Oct. 2009.

[93] W.-H. Chen, S.-H. Hsu, and H.-P. Shen, "Application of SVM and ANN for intrusion detection," *Comput. Oper. Res.*, vol. 32, no. 10, pp. 2617–2634, 2005.

[94] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Machine learning: A review of classification and combining techniques," *Artif. Intell. Rev.*, vol. 26, no. 3, pp. 159–190, 2006.

[95] I. Ahmad, A. Abdullah, A. Alghamdi, and M. Hussain, "Optimized intrusion detection mechanism using soft computing techniques," *Telecommun. Syst.*, vol. 52, no. 4, pp. 2187–2195, 2013.

[96] H.-Y. Huang and C.-J. Lin, "Linear and kernel classification: When to use which?" in *Proc. SIAM Int. Conf. Data Min.*, 2016, pp. 216–224.

[97] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Proc. Adv. Neural Inf. Process. Syst.*, Denver, CO, USA, 2000, pp. 582–588.

[98] S. Owais, V. Snasel, P. Kromer, and A. Abraham, "Survey: Using genetic algorithm approach in intrusion detection systems techniques," in *Proc. 7th IEEE Comput. Inf. Syst. Ind. Manag. Appl. (CISIM)*, Ostrava, Czech Republic, 2008, pp. 300–307.

[99] S. Selvakani and R. S. Rajesh, "Genetic algorithm for framing rules for intrusion detection," *Int. J. Comput. Sci. Netw. Security*, vol. 7, no. 11, pp. 285–290, 2007.

[100] P. Gupta and S. K. Shinde, "Genetic algorithm technique used to detect intrusion detection," in *Proc. 1st Int. Conf. Adv. Comput. Inf. Technol.*, Chennai, India, 2011, pp. 122–131.

[101] O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, "An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks," *Expert Syst. Appl.*, vol. 29, no. 4, pp. 713–722, 2005.

[102] A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," *Data Knowl. Eng.*, vol. 63, no. 2, pp. 503–527, 2007.

[103] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surveys*, vol. 41, no. 3, p. 15, 2009.

[104] A. A. Aburomman and M. B. I. Reaz, "A novel SVM-KNN-PSO ensemble method for intrusion detection system," *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.

[105] H. H. Hosmer, "Security is fuzzy! Applying the fuzzy logic paradigm to the multipolicy paradigm," in *Proc. ACM Workshop New Security Paradigms*, Little Compton, RI, USA, 1993, pp. 175–184.

[106] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, no. 3, pp. 338–353, 1965.

[107] C. Tang, Y. Xiang, Y. Wang, J. Qian, and B. Qiang, "Detection and classification of anomaly intrusion using hierarchy clustering and SVM," *Security Commun. Netw.*, vol. 9, no. 16, pp. 3401–3411, 2016.

[108] S. Raja and S. Ramaiah, "An efficient fuzzy-based hybrid system to cloud intrusion detection," *Int. J. Fuzzy Syst.*, vol. 19, no. 1, pp. 62–77, 2017.

[109] M. Gyanchandani, J. Rana, and R. Yadav, "Taxonomy of anomaly based intrusion detection system: A review," *Int. J. Sci. Res. Publ.*, vol. 2, no. 12, pp. 1–13, 2012.

[110] L. R. Rabiner and B. H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Mag.*, vol. 3, no. 1, pp. 4–16, Jan. 1986.

[111] D. Ariu and G. Giacinto, "HMMPayl: An application of HMM to the analysis of the HTTP payload," in *Proc. WAPA*, 2010, pp. 81–87.

[112] A. Churbanov and S. Winters-Hilt, "Implementing EM and Viterbi algorithms for hidden Markov model in linear memory," *BMC Bioinformat.*, vol. 9, no. 1, p. 224, 2008.

[113] C. Kolias, G. Kambourakis, and M. Maragoudakis, "Swarm intelligence in intrusion detection: A survey," *Comput. Security*, vol. 30, no. 8, pp. 625–642, 2011.

[114] M. Abadi and S. Jalili, "An ant colony optimization algorithm for network vulnerability analysis," *Iran. J. Elect. Elect. Eng.*, vol. 2, no. 3, pp. 106–120, 2006.

[115] C. Blum and X. Li, "Swarm intelligence in optimization," in *Swarm Intelligence*. Cham, Switzerland: Springer, 2008, pp. 43–85.

[116] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2012.

[117] M. Sewell, "Ensemble learning," *Res. Note*, vol. 11, no. 2, pp. 1–18, 2008.

[118] Y. Freund, "Boosting a weak learning algorithm by majority," *Inf. Comput.*, vol. 121, no. 2, pp. 256–285, 1995.

[119] K. Anusha and E. Sathiyamoorthy, "Comparative study for feature selection algorithms in intrusion detection system," *Autom. Control Comput. Sci.*, vol. 50, no. 1, pp. 1–9, 2016.

[120] D. S. Kim and J. S. Park, "Network-based intrusion detection with support vector machines," in *Information Networking. ICOIN 2003* (LNCS 2662), H. K. Kahng. Heidelberg, Germany: Springer, 2003, pp. 747–756.

[121] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems," in *Proc. ACM Symp. Appl. Comput.*, Nicosia, Cyprus, 2004, pp. 420–424.

[122] Y. Bouzida and F. Cuppens, "Neural networks vs. decision trees for intrusion detection," in *Proc. IEEE/IST Workshop Monitor. Attack Detection Mitigation (MonAM)*, vol. 28. Tübingen, Germany, 2006, p. 29.

[123] C. Zhang, J. Jiang, and M. Kamel, "Intrusion detection using hierarchical neural networks," *Pattern Recognit. Lett.*, vol. 26, no. 6, pp. 779–791, 2005.

[124] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," *J. Netw. Comput. Appl.*, vol. 28, no. 2, pp. 167–182, 2005.

[125] W. Wang and R. Battiti, "Identifying intrusions in computer networks with principal component analysis," in *Proc. IEEE 1st Int. Conf. Availability Rel. Security (ARES)*, 2006, p. 8.

[126] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *J. Int. J. Very Large Data Bases*, vol. 16, no. 4, pp. 507–521, 2007.

[127] Y. Li, B.-X. Fang, L. Guo, and Y. Chen, "TCM-KNN algorithm for supervised network intrusion detection," in *Proc. Pac. Asia Conf. Intell. Security Informat.*, Chengdu, China, 2007, pp. 141–151.

[128] Y. Chen, A. Abraham, and B. Yang, "Hybrid flexible neural-tree-based intrusion detection systems," *Int. J. Intell. Syst.*, vol. 22, no. 4, pp. 337–352, 2007.

[129] C. Xiang, P. C. Yong, and L. S. Meng, "Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees," *Pattern Recognit. Lett.*, vol. 29, no. 7, pp. 918–924, 2008.

[130] X. Tong, Z. Wang, and H. Yu, "A research using hybrid RBF/Elman neural networks for intrusion detection system secure model," *Comput. Phys. Commun.*, vol. 180, no. 10, pp. 1795–1801, 2009.

[131] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, "Intrusion detection using fuzzy association rules," *Appl. Soft Comput.*, vol. 9, no. 2, pp. 462–469, 2009.

[132] G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using artificial neural networks and fuzzy clustering," *Expert Syst. Appl.*, vol. 37, no. 9, pp. 6225–6232, 2010.

[133] F. Amiri, M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *J. Netw. Comput. Appl.*, vol. 34, no. 4, pp. 1184–1199, 2011.

[134] S.-J. Horng *et al.*, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Syst. Appl.*, vol. 38, no. 1, pp. 306–313, 2011.

[135] D. Boughaci, M. D. E. Kadi, and M. Kada, "Fuzzy particle swarm optimization for intrusion detection," in *Proc. Int. Conf. Neural Inf. Process.*, Doha, Qatar, 2012, pp. 541–548.

[136] S.-W. Lin, K. C. Ying, C.-Y. Lee, and Z.-J. Lee, "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection," *Appl. Soft Comput.*, vol. 12, no. 10, pp. 3285–3290, 2012.

[137] S. S. S. Sindhu, S. Geetha, and A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 129–141, 2012.

[138] Y. Li *et al.*, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 424–430, 2012.

[139] A. Chandrasekhar and K. Raghuveer, "Intrusion detection technique by using k-means, fuzzy neural network and SVM classifiers," in *Proc. IEEE Int. Conf. Comput. Commun. Informat. (ICCCI)*, Coimbatore, India, 2013, pp. 1–7.

[140] S. Kumar and A. Yadav, "Increasing performance of intrusion detection system using neural network," in *Proc. Int. Conf. Adv. Commun. Control Comput. Technol. (ICACCCT)*, 2014, pp. 546–550.

[141] F. Kuang, W. Xu, and S. Zhang, "A novel hybrid KPCA and SVM with GA model for intrusion detection," *Appl. Soft Comput.*, vol. 18, pp. 178–184, May 2014.

[142] W. Feng, Q. Zhang, G. Hu, and J. X. Huang, "Mining network data for intrusion detection through combining SVMs with ant colony networks," *Future Gener. Comput. Syst.*, vol. 37, pp. 127–140, Jul. 2014.

[143] L. Koc, T. A. Mazzuchi, and S. Sarkani, "A network intrusion detection system based on a hidden Naïve Bayes multiclass classifier," *Expert Syst. Appl.*, vol. 39, no. 18, pp. 13492–13500, 2012.

[144] W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "Cann: An intrusion detection system based on combining cluster centers and nearest neighbors," *Knowl. Based Syst.*, vol. 78, pp. 13–21, Apr. 2015.

[145] P. V. Amoli, T. Hamalainen, G. David, M. Zolotukhin, and M. Mirzamohammad, "Unsupervised network intrusion detection systems for zero-day fast-spreading attacks and botnets," *Int. J. Digit. Content Technol. Its Appl.*, vol. 10, no. 2, pp. 1–13, 2016.

[146] G. Kumar and K. Kumar, "A multi-objective genetic algorithm based approach for effective intrusion detection using neural networks," in *Intelligent Methods for Cyber Warfare* (Studies in Computational Intelligence), vol. 563, R. Yager, M. Reformat, and N. Alajlan, Eds. Cham, Switzerland: Springer, 2015, pp. 173–200.

[147] A. N. Toosi and M. Kahani, "A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers," *Comput. Commun.*, vol. 30, no. 10, pp. 2201–2212, 2007.

[148] S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, and F. Herrera, "On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems," *Expert Syst. Appl.*, vol. 42, no. 1, pp. 193–202, 2015.

[149] W. Yassin, N. I. Udzir, Z. Muda, and M. N. Sulaiman, "Anomaly-based intrusion detection through k-means clustering and Naives Bayes classification," in *Proc. 4th Int. Conf. Comput. Informat. (ICOCI)*, 2013, pp. 298–303.

[150] K. K. Gupta, B. Nath, and R. Kotagiri, "Layered approach using conditional random fields for intrusion detection," *IEEE Trans. Depend. Secure Comput.*, vol. 7, no. 1, pp. 35–49, Jan./Mar. 2010.

[151] M. S. I. Mamun, A. A. Ghorbani, and N. Stakhanova, "An entropy based encrypted traffic classifier," in *Proc. Int. Conf. Inf. Commun. Security*, Beijing, China, 2015, pp. 282–294.

[152] D. Bhamare, T. Salman, M. Samaka, A. Erbad, and R. Jain, "Feasibility of supervised machine learning for cloud security," in *Proc. IEEE Int. Conf. Inf. Sci. Security (ICISS)*, Pattaya, Thailand, 2016, pp. 1–5.

[153] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Tehran, Iran, 2016, pp. 139–144.

[154] M. N. Chowdhury, K. Ferens, and M. Ferens, "Network intrusion detection using machine learning," in *Proc. Int. Conf. Security Manag. (SAM)*, 2016, pp. 1–7.

[155] N. Moustafa and J. Slay, "A hybrid feature selection for network intrusion detection systems: Central points," in *Proc. 16th Aust. Inf. Warfare Conf.*, 2015, pp. 1–10.

[156] A. M. Chandrasekhar and K. Raghuveer, "Intrusion detection technique by using k-means, fuzzy neural network and SVM classifiers," in *Proc. IEEE Int. Conf. Comput. Commun. Informat. (ICCCI)*, Coimbatore, India, 2013, pp. 1–7.

[157] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-based application layer DoS attacks on Web servers in the presence of sampling," *Comput. Netw.*, vol. 121, pp. 25–36, Jul. 2017.

[158] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on SVM with feature augmentation," *Knowl. Based Syst.*, vol. 136, pp. 130–139, Nov. 2017.

[159] Akashdeep, I. Manzoor, and N. Kumar, "A feature reduced intrusion detection system using ANN classifier," *Expert Syst. Appl.*, vol. 88, pp. 249–257, Dec. 2017.

[160] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.

[161] S. M. H. Bamakan, H. Wang, T. Yingjie, and Y. Shi, "An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization," *Neurocomputing*, vol. 199, pp. 90–102, Jul. 2016.

[162] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications* (Mathematics and Its Applications), vol. 37, P. J. van Laarhoven and E. H. Aarts, Eds. Dordrecht, The Netherlands: Springer, 1987, pp. 7–15.

[163] A. A. Olusola, A. S. Oladele, and D. O. Abosede, "Analysis of KDD '99 intrusion detection dataset for selection of relevance features," in *Proc. World Congr. Eng. Comput. Sci.*, vol. 1, 2010, pp. 20–22.

[164] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, "Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets," in *Proc. 3rd Annu. Conf. Privacy Security Trust*, 2005, pp. 1–6.

[165] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "Securing virtual machines from anomalies using program-behavior analysis in cloud environment," in *Proc. IEEE 18th Int. Conf. High Perform. Comput. Commun.*, 2016, pp. 991–998.
[166] (2016). *Weka 3.8.1: Data Mining Software in Java*. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/
[167] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[168] Google. (2017). *Installing Tensorflow*. [Online]. Available: https://www.tensorflow.org/install/
[169] KNIME.com. (2017). *Knime 3.4.1: Download Knime Analytics Platform & SDK*. [Online]. Available: https://www.knime.com/downloads
[170] RapidMiner. (2017). *Real Data Science, Fast and Simple (Stable Release 7.5)*. [Online]. Available: https://rapidminer.com/
[171] E. Achtert, H.-P. Kriegel, and A. Zimek, "ELKI: A software system for evaluation of subspace clustering algorithms," in *Scientific and Statistical Database Management* (LNCS 5069), B. Ludäscher and N. Mamoulis, Eds. Heidelberg, Germany: Springer, 2008, pp. 580–585.
[172] University of Waikato. (2014). *MOA (Massive Online Analysis)*. [Online]. Available: https://moa.cms.waikato.ac.nz/
[173] A. Bifet and G. D. F. Morales, "Big data stream learning with SAMOA," in *Proc. IEEE Int. Conf. Data Min. Workshop (ICDMW)*, Shenzhen, China, 2014, pp. 1199–1202.
[174] X. Meng *et al.*, "MLlib: Machine learning in apache spark," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1235–1241, 2016.
[175] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends® Signal Process.*, vol. 7, no. 3–4, pp. 197–387, 2014.
[176] E. Aminanto and K. Kim, "Deep learning in intrusion detection system: An overview," in *Proc. Int. Res. Conf. Eng. Technol.*, 2016, pp. 1–12.
[177] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in *Proc. 10th IEEE Int. Conf. Malicious Unwanted Softw. (MALWARE)*, 2015, pp. 11–20.
[178] Z. Wang, "The applications of deep learning on traffic identification," presented at the BlackHat, Las Vegas, NV, USA, 2015, pp. 1–10.
[179] Y. Li, R. Ma, and R. Jiao, "A hybrid malicious code detection method based on deep learning," *Int. J. Softw. Eng. Appl.*, vol. 9, no. 5, pp. 205–216, 2015.
[180] W. Yan and L. Yu, "On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach," in *Proc. Annu. Conf. Prognostics Health Manag. Soc.*, 2015, pp. 1–8.
[181] N. Gao, L. Gao, Q. Gao, and H. Wang, "An intrusion detection model based on deep belief networks," in *Proc. IEEE 2nd Int. Conf. Adv. Cloud Big Data (CBD)*, Huangshan, China, 2014, pp. 247–252.
[182] W. Jung, S. Kim, and S. Choi, "Poster: Deep learning for zero-day flash malware detection," in *Proc. 36th IEEE Symp. Security Privacy*, 2015, pp. 1–2.
[183] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[184] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted Boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, Dec. 2013.
[185] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: Android malware characterization and detection using deep learning," *Tsinghua Sci. Technol.*, vol. 21, no. 1, pp. 114–123, 2016.
[186] Y. Wang, W.-D. Cai, and P.-C. Wei, "A deep learning approach for detecting malicious JavaScript code," *Security Commun. Netw.*, vol. 9, no. 11, pp. 1520–1534, 2016.
[187] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, "Hybrid intelligent intrusion detection scheme," in *Soft Computing in Industrial Applications* (Advances in Intelligent and Soft Computing), vol. 96, A. Gaspar-Cunha, R. Takahashi, G. Schaefer, and L. Costa, Eds. Heidelberg, Germany: Springer, 2011, pp. 293–303.
[188] S. Seok and H. Kim, "Visualized malware classification based-on convolutional neural network," *J. Korea Inst. Inf. Security Cryptol.*, vol. 26, no. 1, pp. 197–208, 2016.
[189] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *Proc. 8th IEEE Int. Conf. Commun. Softw. Netw. (ICCSN)*, Beijing, China, 2016, pp. 581–585.
[190] K. Malialis and D. Kudenko, "Distributed response to network intrusions using multiagent reinforcement learning," *Eng. Appl. Artif. Intell.*, vol. 41, pp. 270–284, 2015.
[191] A. Servin and D. Kudenko, "Multi-agent reinforcement learning for intrusion detection," in *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning* (LNCS 4865), K. Tuyls, A. Nowe, Z. Guessoum, and D. Kudenko, Eds. Heidelberg, Germany: Springer, 2008, pp. 211–223.
[192] X. Xu and T. Xie, "A reinforcement learning approach for host-based intrusion detection using sequences of system calls," in *Advances in Intelligent Computing. ICIC 2005* (LNCS 3644). Heidelberg, Germany: Springer, 2005, pp. 995–1003.
[193] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.
[194] UNM. (1998). *UNM Dataset*. [Online]. Available: http://www.cs.unm.edu/immsec/systemcalls.htm
[195] E. Hodo, X. J. A. Bellekens, A. Hamilton, C. Tachtatzis, and R. C. Atkinson, "Shallow and deep networks intrusion detection system: A taxonomy and survey," *ACM Survey*, 2017. [Online]. Available: http://arxiv.org/abs/1701.02145
[196] W. Qiang and Z. Zhongli, "Reinforcement learning model, algorithms and its application," in *Proc. IEEE Int. Conf. Mechatronic Sci. Elect. Eng. Comput. (MEC)*, 2011, pp. 1143–1146.
[197] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI*, 2016, pp. 2094–2100.

Preeti Mishra (M'14) received the Ph.D. degree in computer science and engineering from the Malaviya National Institute of Technology Jaipur, India, in 2017, under the supervision of Dr. Emmanuel S. Pilli and Prof. V. Varadharajan. She is currently an Associate Professor with Graphic Era University, Dehradun, India. She has been a Visiting Scholar at Macquarie University, Sydney, NSW, Australia, in 2015. Her area of interest includes Cloud security, E-mail security, Network security, and IoT.

Vijay Varadharajan is the Global Innovation Chair Professor with the University of Newcastle, Australia and the Director of the Advanced Cyber Security Research Centre. He has published over 380 papers in international journals and conferences, ten books on information technology, security, networks, and distributed systems, and has held three patents. He has been/is on the Editorial Board of several journals including *ACM Transactions on Information and System Security*, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and the IEEE TRANSACTIONS ON CLOUD COMPUTING.

Uday Tupakula (M'12) received the Ph.D. degree in 2016, under the supervision of Prof. Varadharajan. He is a Senior Lecturer with the University of Newcastle, Australia. He has 75 publications in different research areas such as network security, DDoS attacks, MANET security, and secure virtual systems. He is a member of BCS and ACM.

Emmanuel S. Pilli (SM'16) received the Ph.D. degree from IIT Roorkee, Roorkee, in 2012. He is currently an Associate Professor with the Malaviya National Institute of Technology, Jaipur, India. He has 20 years of teaching, research, and administrative experience. His areas of interest include Security and Forensics, Cloud computing, Big data, and IoT. He is also a Senior Member of ACM and CSI and actively involved in Cloud Computing Innovation Council of India, NIST Cloud Forensic Workgroup.