IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.DOI

A Review of Rule Learning Based Intrusion Detection Systems and Their Prospects in Smart Grids

QI LIU, VEIT HAGENMEYER, and HUBERT B. KELLER
Institute for Automation and Applied Informatics, Karlsruhe Institute of Technology, Eggenstein-Leopoldshafen 76344, Germany

Corresponding author: Qi Liu (e-mail: qi.liu@kit.edu).

This work was supported by funding of the Helmholtz Association (HGF) through the Energy System Design (ESD) program and through the Competence Center for Applied Security Technology (KASTEL).

ABSTRACT
Intrusion detection systems (IDS) are commonly categorized into misuse based, anomaly based and specification based IDS. Both misuse based IDS and anomaly based IDS are extensively researched in academia and industry. However, as critical infrastructures including smart grids (SG) may often face sophisticated unknown attacks in the near future, misuse based attack detection techniques will mostly miss their targets. Despite the fact that anomaly based IDS can detect novel attacks, they are not often deployed in industry, mainly owing to high false positive rate and lack of interpretability of trained models. With misuse based IDS' inability to detect unknown attacks and requirement for frequently manually crafting and updating signatures and with anomaly based IDS' bad reputation for high false alarm rate, specification based IDS can be regarded as the most suitable detection engine for cyber-physical systems (CPS) including SG. We argue that specification based IDS especially using rule learning could prove to be the most promising IDS for SG. Intrusion detection rules are learned through rule learning techniques and periodically automatically updated to accommodate dynamic system behaviors in SG. Fortunately, rule learning based IDS can not only detect previously unknown attacks but also achieve higher interpretability, due to symbolic representation of learned rules. It can thus be considered more "trustworthy" from human perspective and further assist human in the loop security operation. The present work provides a systematic and deep analysis of rule learning techniques and their suitability for IDS in SG. Besides, it concludes the most important criteria for learning intrusion detection rules and assessing their quality. This work serves not only as a guide to a number of important rule learning techniques but also as the first survey on their applications in IDS, which indicates their potential opportunities in SG security.

INDEX TERMS Intrusion Detection, Interpretable Machine Learning, Rule Learning, Data Mining, Smart Grid

I. INTRODUCTION

As a complement to security appliances like encryption, authentication, authorization, firewalls and VPN, intrusion detection systems (IDS) are often regarded as the second defense line in the so called defense in depth. The defense in depth concept is referred to a holistic approach that combines several countermeasures implemented in layers to create an aggregated, risk-based security for defending against cyber-security threats [1].

In many cyber-physical systems (CPS) like smart grids (SG), though, the lack of basic security measures like encryption and authentication in their standard communication protocols like Modbus and DNP3 makes industrial networks especially vulnerable. With the raised security attention, efforts are made to entrench security infrastructure, such as by introducing authentication in Secure DNP3 protocol. However, the secure versions of industrial protocols are not always implemented due to various factors like added complexity and varying vendor support [2]. Even if security measures are applied in industrial protocols, it is still necessary to have IDS acting as the second defense line regarding to the defense in depth concept as aforementioned.

A good number of soft computing based methods and solutions have been adapted in IDS in cybersecurity research works to improve detection accuracy and efficiency. Seeking to imitate human intelligence, to improve learning and decision-making processes and thus to solve real-world problems, soft computing as a general term describes a set of optimization techniques including fuzzy logic, artificial neural networks, probabilistic reasoning, association rule mining, genetic algorithms, particle swarm intelligence, ant colony optimization etc. [3]. Often their applications in IDS are inspired by the successful use cases in other scientific fields like medicine, bioinformatics, economics, computer networks etc. What makes them attractive to be applied in intrusion detection is that soft computing techniques are capable of handling uncertain and partially true data, which are oftentimes seen in cybersecurity research field. Inexpensive solutions are achieved with an acceptable trade-off between robustness and forbearance for inaccuracy and partial truth [4]. When used in intrusion detection, soft computing techniques can be complemented by rule based expert knowledge described as a set of IF-THEN rules [5]. An IF-THEN rule is composed of a rule body, i.e., antecedent, and a rule head, i.e., consequent. The rule body or antecedent is represented as a conjunction of conditions, also called as attributes or constraints, which need to be fulfilled to "fire" a rule. The rule head or consequent can be "attack", "no-attack" or "unknown."

In specification based IDS, specifications can be generated either manually from human experts or automatically from rule learning techniques. The rules or policies in the majority of current specification based IDS are created manually by human experts, and they can be based on protocol specifications like in [6] and [7]. If all the allowable actions or behaviors of a system can be known and described easily beforehand, then the specifications can be formulated by human experts and implemented in IDS without altering it afterwards. However, oftentimes, the set of possible benign system behaviors or action sequences can not be fully ascertained before its real-world deployment and/or should be adjusted periodically after the deployment. That is to say, generating specifications from human experts can prove to be time-consuming and error-prone. Nevertheless, we see a great laborsaving potential in rule learning techniques that are yet to be fully explored for practical use cases in IDS. Hence, the present work mainly investigates specification based IDS making use of rule learning techniques. As one of the core technologies in machine learning, the topic of rule learning is large and complicated. It is worth mentioning that rules are applied in a broad variety of machine learning activities in different areas and for different purposes, which sometimes leads to invention of totally different terminologies [8].

Note that in the literature, on the one hand, specification based IDS are sometimes considered as misuse based IDS, since they also use rules for attack detection, but with the capability of detecting unknown attacks. On the other hand, specification based IDS are seen as more strict anomaly based IDS for detecting abnormal system behaviors, as per [9]. Besides, specification based IDS are also referred to as model based IDS [7] or behavior based IDS [10]. We advocate that the fundamental difference between anomaly based techniques and rule learning based techniques is that the former ones directly model training data to detect data point deviation and the latter ones try to infer rules from training data to model the system behaviors and thus to detect policy violation. Due to the fact that these inferred rules or policies are often human understandable, rule learning based approaches can contribute to higher interpretability of decision-making process and probably better situation awareness as well. Interpretability is described by Miller [11] as the degree to which a human can understand the cause of a decision. Moreover, rules are preferred by security analysts as allowing them to easily express their domain knowledge in simple conditions [12].

In spite of the current shift from learning logical concept representations to statistical learning algorithms in machine learning research, rule learning algorithms are still applied in a wide range of areas such as Semantic Web, whose knowledge representation process is supported and automated by rule learning techniques [13]. Likewise, we observe that the current research trend in machine learning based IDS is running with bias towards deep learning based approaches, which however suffer from their arguably best-known drawback: lack of explainability[1] due to their black-box nature. Moreover, we believe that rule learning based IDS are currently somehow understudied and their potentials especially for CPS like SG are yet to be explored with more efforts. Through this work, we aim to attract more attention from security researchers into rule learning techniques to develop more explainable and hence more practical machine learning based IDS.

The remainder of this paper is organized as follows: Section II overviews a number of existing IDS surveys with various foci, and highlights the distinction between our work and the previous ones. Section III gives an introduction into IDS and numerous categorization criteria. Section IV presents an overview of SG, communication technologies in SG and attack space in SG. Section V summarizes a number of rule learning techniques, whose relationships and differences are highlighted and explained in an easily understandable way. Section VI emphasizes the importance of dataset, feature engineering and performance metrics. Section VII discusses various applications of rule learning based IDS. Section VIII presents the existing challenges and prospects of intrusion detection research. Section IX concludes this work and points out our future directions.

[1]Given that explainability and interpretability are often used interchangeably in machine learning community due to being closely related, they do not differ from each other in this work either. However, it is worth noting that they are distinguishable in which explainable models are interpretable by default, but the reverse does not always hold true [14]. Simply put, a model can be understood even better if it is explainable rather than interpretable.

II. RELATED WORK

With plenty of studies performed to review the current state of intrusion detection systems, the foci vary from one to another in analyzing, comparing and summarizing the investigated intrusion detection techniques and identifying research gaps and future research directions. For instance, Aburomman *et al.* [15] provide an overview of IDS based on ensemble and hybrid classifiers considering both homogeneous and heterogeneous types of ensemble methods, whereas Zhou *et al.* [16] present a review of collaborative IDS against coordinated attacks like distributed denial-of-service (DDoS) attacks. Arshad *et al.* [17] compare the current IDS laying emphasis on computational overhead, energy consumption and privacy implications and point out that these aspects are yet to be highly considered in future IDS research. Berman *et al.* [18] give a comprehensive review of deep learning methods applied for cybersecurity. A survey of IDS only considering host-based approaches is presented by Bridges *et al.* [19]. Buczak *et al.* [20] present a summary of IDS leveraging data mining and machine learning approaches and propose that the methods for fast incremental learning should be further exploited. Chandola *et al.* [21] provide a very comprehensive overview of anomaly detection covering a plethora of applied techniques in the literature.

Mitchell *et al.* [22] investigate proposed intrusion detection approaches for CPS, and discuss merits and drawbacks of various intrusion detection techniques when applying to CPS. They indicate that physical process monitoring, closed control loops, attack sophistication and legacy technology represent the uniqueness of intrusion detection in CPS. Moreover, they emphasize that more efforts ought to be given to specification based and federated IDS. Whereas Tong *et al.* [9] present a survey of IDS concerning only advanced metering infrastructure (AMI) of SG, Grammatikis *et al.* [23] provide a very comprehensive overview of IDS for SG ecosystems and subsystems. Besides, Grammatikis *et al.* point out that in the literature no IDS is demonstrated specifically for protecting microgrids of SG yet.

Furthermore, a handful of surveys only targeting IDS datasets are also performed, e.g., [24]. A summary of these previous works is provided in Table 1. However, as showed in Table 1, none of them explicitly investigates rule learning based IDS. To the best of our knowledge, no systematic review of IDS based on rule learning techniques was performed in the research community previously. Hence we aim at providing a systematic and deep analysis of rule learning techniques and their suitability for IDS in SG. Besides, we conclude the most important criteria for assessing quality of learned intrusion detection rules. Apart from giving a gentle introduction to intrusion detection systems and smart grids, we also provide the first survey focusing on rule learning based IDS. As mentioned in Section I, this is of great importance in terms of interpretability and practicality of developed intrusion detection methods.

TABLE 1. Summary of related existing surveys

References	Main objectives
Aburomman *et al.* [15]	IDS based on ensemble and hybrid classifiers
Zhou *et al.* [16]	Collaborative IDS against coordinated attacks
Arshad *et al.* [17]	Computational overhead, energy consumption and privacy implications of IDS for IoT
Berman *et al.* [18]	Deep learning methods applied for cybersecurity
Bridges *et al.* [19]	IDS leveraging host data
Buczak *et al.* [20]	IDS based on data mining and machine learning approaches
Chandola *et al.* [21]	Anomaly detection techniques in general
Mitchell *et al.* [22]	IDS for cyber-physical systems
Tong *et al.* [9]	IDS for advanced metering infrastructure
Grammatikis *et al.* [23]	IDS for SG ecosystems and subsystems
Ring *et al.* [24]	Network-based IDS datasets

III. INTRUSION DETECTION SYSTEM AND ITS CATEGORIZATION

As a complementary security application in the defense in depth suite, an IDS is a program or device applied to monitor a system or network and thus to detect intrusive activities against it. Any suspicious events are flagged and reported to an administration system. Although IDS are often associated with firewall, the main difference lays in that a firewall is configured with a set of static (ordered) rules to block some system activities or network connections and hence to prevent intrusions without giving much efforts to examine them. However, IDS can also be integrated in a firewall, referred to as next-generation firewall (NGFW).

As it can be found in the literature, there are a plethora of ways to categorize IDS. Yet it is often seen that the categorization criteria are not explicitly mentioned. Moreover, a tree structure is usually used to show the taxonomy of IDS. This kind of depiction, though, can sometimes be a little confusing, since differently categorized IDS are often not mutually exclusive and this aspect can hardly be highlighted in a tree structure. Thus we prefer using plain text to categorize IDS and meanwhile always pointing out the categorization criterion.

A. CATEGORIZATION BASED ON TARGET SYSTEM

IDS can be categorized into host-based IDS and network-based IDS according to the target system being monitored and protected. Whereas host-based IDS aims at only monitoring a single host/computer, the observation scope of network-based IDS is an entire or part of network with multiple hosts.

1) Host-based IDS

Host-based IDS is a computer program installed on individual devices. It focuses on activities in a device and strives to detect intrusions usually by analyzing audit trails or system logs produced by the host operating system. Host-based IDS is deemed to be better suitable for detecting attacks for a particular device.

2) Network-based IDS

Network-based IDS, also referred to as IDS sensor, is often a dedicated hardware with a collection of special software.

It analyzes packets captured directly from a network to catch possible attacks against hosts in the network. Hence its location needs to be carefully considered for being able to catch all important network traffics. It is usually put at a strategic point in a network via a TAP (test access point) or SPAN (switch port analyzer).

B. CATEGORIZATION BASED ON DATA

IDS can also be categorized on the basis of input data type. Although audit logs are mostly associated with host-based IDS, and network packets are mostly associated with network-based IDS, these data types do not indicate the demarcation between host-based IDS and network-based IDS.

1) Audit logs based IDS

Audit logs are the historical reports of system behaviors and provide an invaluable view into the current and past system states. The trails can be analyzed to help determine what happened, thus to detect intrusions. Most software in existence may include some logging mechanisms. The other way around, alerts generated from IDS themselves are also a form of audit logs for tracing the source of attacks. Audit logs can not only be used in host-based IDS, often in the form of system calls, but also be taken as data source in network-based IDS. For example in [25], event logs generated from network communication are applied for detecting attacks against hosts on the network.

2) Network flow based IDS

Network flow based IDS, also referred to as network connection-oriented IDS, utilize data of network layer and transport layer of the OSI model. This kind of IDS is primarily amenable for detecting attacks at network level, such as denial-of-service attacks, port scanning. Packet data can be used not only in network-based IDS but also in host-based IDS. An example is stack based IDS, which operate directly on the TCP/IP stack and pull the packets from the stack before the host operating system.

3) Packet payload based IDS

Packet payload often means application layer data. As per [26], in the modern attack landscape, vulnerabilities at application layer are the main targets of attackers. Most of these attacks appear to be normal when only considering packet header attributes, however, may significantly differ from legitimate traffic if packet payloads are checked. That is to say, solely relying on network flow data may no longer be a viable solution and packet payload based IDS is essential for attack detection. Moreover, based on how data are utilized, IDS can be categorized into stateless and stateful IDS, in which only stateful IDS leverage sequential data ordered by timestamp.

C. CATEGORIZATION BASED ON DETECTION TECHNIQUE

Detection technique is probably the most used IDS categorization criterion. Each technique has its own merits and drawbacks, and none of them is perfectly suitable for every situation. Detection techniques can also be combined to create hybrid ones.

1) Misuse based IDS

Misuse based IDS, often referred to as signature based IDS, rely on a database of signatures/patterns of known attacks. When deployed, the IDS try to match analyzed data against these signatures. The underlying assumption is that the characteristics of intrusions are mapped in these signatures, and matches found signify malicious activities. Although misuse based IDS are accredited for low false alarm rate and are widely in use, it is often pointed out that misuse based IDS are easy to circumvent, due to the fact that it is usually possible to modify the syntax of an attack without changing its semantics [26].

2) Anomaly based IDS

The first anomaly detection model was introduced by Denning [27] as complementary to misuse based detection methods. Statistical models describing normal behaviors are built to catch significant deviations. Anomaly based IDS have an underlying assumption that normal behaviors can be statistically modeled and any deviations from baseline models can be seen as intrusive actions. As per [21], anomalies fall into point anomalies, contextual anomalies and collective anomalies. As the simplest anomaly type and the focus of anomaly detection research, point anomalies represent individual data instances. A contextual anomaly is regarded as conditional anomaly in which each data instance has contextual attributes and behavior attributes. A collective anomaly is associated with sequence, graph or spatial data.

3) Specification based IDS

The concept of specification based intrusion detection was firstly proposed by Ko [28]. It is based on the following assumption: profiles or rules for attack detection can be specified by using expert knowledge and/or by learning from only limited amount of data. Violation against constructed normal profiles or rules is considered as malicious. Specification based IDS differ from their anomaly based counterparts from the aspect that they try to construct declarative knowledge instead of a set of procedures that does not have any contextual meaning, i.e., from a human-reasoning perspective [29]. We believe that this aspect is especially useful for intrusion detection in SG, since specifications may allow the description of extreme variations in regular operation of SG. Nevertheless, developing a good specification can be regarded as a hard task, since it requires significant insight into complex programs and continuous analysis for new program revisions [29].

4) Hybrid IDS

As it can be found in the literature, hybrid IDS are a current hot research topic. The underlying assumption is that the best detection performance can perhaps only be achieved by combining different techniques in a harmonized way and aggregating their merits. For example in [6], the authors combine specification based IDS with anomaly detection methods to ease the task of constructing specification and to reduce false positive rate. In [30], a misuse based IDS complemented with anomaly detection is proposed to enhance detection rate of both known and unknown attacks while maintaining low false alarm rate.

One should not forget that IDS can also be categorized via many other criteria not mentioned above. For example, based on architecture, IDS can be categorized into standalone IDS and distributed IDS, which include centralized and decentralized IDS.

IV. SMART GRIDS AND ATTACK SPACE

A smart grid can be considered as the fusion of two major networks, i.e., a power network and a communication network. While the massive integration of renewable energy contributes to the main novelty of power network in SG, the rapid evolution of communication network in SG is supported by numerous emerging information and communications technologies (ICT). Backed by the aggregation of a rising number of communication technologies, the entire process in SG, including energy generation, transmission, distribution and consumption, can be optimized to reach higher efficiency, transparency and lower cost both for energy providers and consumers. However, the increased connectivity also brings many risks to SG assets.

A. SMART GRIDS ARCHITECTURE

SG are by no means a single system, but rather a very complex interconnection of multiple systems, also referred to as SG subsystems. Most of these individual SG subsystems per se are also complex. Though, it is important to understand the architecture of SG in order to understand the information communication flow, and thus to protect SG against malicious actors. It is beneficial to possess a good grasp of SG subsystems and their own components. A good comprehension of how various components and subsystems are deployed and interconnected helps identify vulnerable sections and potential attack space in SG [31].

Due to the modernization of existing power generation, transmission, distribution and metering infrastructures, the very complex SG architecture is currently usually broken down into following subsystems: bulk power generation systems, microgrids, SCADA (supervisory control and data acquisition) systems, transmission systems with synchrophasors, distribution substations, and advanced metering infrastructure (AMI). From an intrusion detection viewpoint, understanding SG architecture is helpful for developing efficient detection techniques, as different SG subsystems have different characteristics and some techniques are more suitable than the others in a given subsystem.

B. ICT IN SMART GRIDS

The "smart" features of SG heavily rely on advanced information and communications technologies. The proliferation of ICT in SG can be very different in various SG subsystems, and hence the ICT landscapes look different in individual subsystems. We group these subsystems into four sections according to their ICT similarities.

1) ICT in power generation systems/microgrids/SCADA

As the backbone of SG, power generation systems, microgrids and SCADA systems are where industrial control processes are mostly involved. The ICT in this section share a lot of similarities with ICT in industrial networks. In these networks, communication needs to fulfill the requirements of very high reliability, availability and very low latency. As per [2], the communication protocols in industrial networks can be categorized into fieldbus protocols and backend protocols. Fieldbus protocols represent a broad variety of protocols that are commonly applied in process and control level. They are used to connect process devices to control devices, e.g., field sensors to PLC (programmable logic controller), as well as to connect control devices to supervisory devices, e.g., PLC to HMI (human-machine interface). Prevalent fieldbus protocols include Modbus RTU, Modbus TCP, DNP3, PROFIBUS, PROFINET, industrial Ethernet etc. Backend protocols include the communication protocols deployed on or above supervisory level, which provide efficient intersystems communication, e.g., between operation control centers. Popular backend protocols are OPC (open process communication), ICCP (inter-control center communications protocol) etc. [2].

2) ICT in AMI

As the main enabler of "smart" grids, AMI provides bidirectional communication between energy consumers and providers for monitoring and demand-response system. In AMI, a broad variety of new communication technologies emerge and constantly evolve, which contributes to the integration of a great number of intelligent appliances into SG. When analyzing information flow, AMI is often broken down into LAN (HAN/IAN/BAN), NAN and WAN, which represent different area networks regarding to their geographical coverage. Whereas smart meters serve as the demarcation between LAN and NAN, metering data collectors mark the boundary between NAN and WAN. Power line communication (PLC), as the predominant smart metering technology in the EU and China due to no need of extra wiring [32], can be used through an entire AMI. Whereas narrowband PLC is applied for communication between power consumers and smart meters as well as between smart meters and metering data collectors, broadband PLC is employed for communication between metering data collectors and metering head-end systems. Nevertheless, a plethora of other communication technologies including wired and wireless ones are broadly deployed in AMI as well, of which all have their own merits subject to use cases. For example, wireless technologies are the most popular options for automatic meter reading in the USA, among which cellular networks and radio frequency mesh networks are widely in use [32]. The ICT in AMI advances probably most quickly in HAN/IAN/BAN (home/industry/business area network). Prevalent communication technologies in these networks include IEEE 802.15.4, Zigbee, Z-Wave, Wi-Fi HaLow, Ethernet, Bluetooth Low Energy, PLC [32] [33] [34].

3) ICT in transmission systems

One of the most frequently mentioned ICT in transmission systems is the IEEE C37.118 standard, which covers the synchronization and communication of phasor measurements in transmission systems. By allowing multiple measurements to be synchronized together, the IEEE C37.118 standard is essential for obtaining overall power quality assessments, since an isolated phasor measurement does not provide much value. IEEE C37.118 standard defines the real-time communications of measurements from phasor measurement unit (PMU) to phasor data concentrator (PDC) with synchronization realized by tagging each measurement with an absolute time reference, such as a GPS clock [31]. However, C37.118 represents a similar challenge as other industrial control protocols, i.e., the lack of inherent security. Phasor measurements can be easily intercepted or manipulated if PMU, PDC and their communications are not sufficiently protected [35].

4) ICT in transmission/distribution substations

The IEC 61850 standard and IEC 62351 standard have gained a wide acceptance in transmission/distribution substations. While the IEC 61850 provides a number of useful specifications, e.g., data modeling, reporting, data transfer and command capability, as well as event messaging using GSE (GOOSE/GSSE), it does not include its own security specifications. The IEC 62351, on the contrary, provides security specifications for substation communications and is broken down into several parts. The IEC 62351-6 is responsible for the security of IEC 61850 messaging and is of particular interest [36] [2].

C. ATTACK SPACE IN SMART GRIDS

The great reliance of SG on ICT, however, can potentially expose all elements of SG to malicious attacks. That is to say, SG are facing immense potential threats, which could affect the SG deployment and growth. The intentions of attackers can vary from one to another, including data theft, financial gain, service disruption and assets sabotage. For example, the bidirectional communication in AMI, as inherent criticality and availability of AMI, is a high-potential target for large scale attacks, which may cause regional blackouts and hence harmful consequences [37]. Moreover, a manipulated reading of a synchrophasor might initiate a faulty state. The manipulation of substation automation systems might cause local loss of service. The attacks leveraging the interconnection mechanisms of SG subsystems might be the most dangerous ones [31]. That is to say, attacks against one subsystem could be utilized as staged attacks against other subsystems, since those subsystems are mutually dependent. For example, a successful intrusion on distribution substation could be leveraged to further attack AMI, owing to the close interdependence between power demand-response system of AMI and distribution substations where metering data collectors are often located [31].

V. RULE LEARNING TECHNIQUES

In general, rule learning techniques are categorized into descriptive rule learning and predictive rule learning. Whereas descriptive rule learning focuses on only finding patterns in a given dataset without considering evaluation on new data samples, predictive rule learning produces rules capable of generalizing to new data samples [13] [8]. For intrusion detection, though, we are almost exclusively interested in predictive rule learning. Unlike descriptive rule learning, predictive rule learning often confronts two types of problems, i.e., multiple rules fire on the same new example, and no rule fires on a new example. In the former case, more than one rule firing on a single example can cause contradiction, and this conflict is resolved either by preferring rules with higher importance or by extracting a separate rule set for handling contradictory predictions. Like in expert systems [38], top-level control parameters are used to handle rule contradictions. The second problem is tackled either by a predefined default rule favoring the majority class or by more complex algorithms finding the closest rule [13].

We conclude that the following criteria should be considered when assessing the quality of a derived rule set for intrusion detection:

Rule correctness: This explains how often the extracted rules hold true on existing datasets, which is mostly reflected in detection rate and false positive rate.

Rule coverage: This evaluates how complete a rule set is and to what extent the derived rules can be generalized for prediction on new data samples.

Rule redundancy: This states the degree of compactness of a rule set and whether there exists overlapping between rules. The goal is to eliminate redundant rules in a rule set.

Rule length: This refers to the number of conditions/attributes in a rule body/antecedent, which should be as small as possible while achieving the same rule correctness. Removing redundant conditions in a rule body is beneficial for rule interpretability and hence its applicability as well.

Rule set size: This indicates the number of rules in a rule set. Sometimes for the sake of short processing time, it can be necessary to reduce the number of rules for time critical applications, even if it slightly decreases the detection accuracy, i.e., trade-off between detection efficiency and detection accuracy.

Rule set freshness: This denotes how much rules get updated periodically after receiving new data instances, which reflects the incremental learning ability, i.e., adaptability to environmental changes.

It is worth noting that all these criteria can be integrated into a fitness function for learning process, but must be carefully weighted due to their potential interdependence and contradictions.

The rule learning techniques presented in the following can be referred to either as rule learning strategies, i.e., how to induce rules, or as rule representation strategies, i.e., how to construct and represent rules and thus to accelerate rule induction process. Both rule learning strategies and rule representation strategies are of critical importance for effectively alleviating the oftentimes burdensome process of extracting useful representative rules. Although the following methods are discussed separately, they are often associated or overlapped with each other and frequently combined for rule induction.

A. DECISION RULE

As the most expressive and comprehensible knowledge representation, decision rules are constantly exploited with various approaches. Two very prevalent learning strategies are divide-and-conquer, i.e., recursively partitioning feature space such as decision tree, and separate-and-conquer, i.e., repeated rule learning and removal of covered data points [39]. Decision rules induced from dataset can be represented either as an unordered rule set or as an ordered rule set, also known as decision list. Whereas with an unordered rule set all rules have to be successively tried to make the democratized prediction for a new instance, a decision list has an order for rules, of which only the first rule satisfying a new instance matters. Decision lists inherently solve the first problem faced by predictive rule learning as aforementioned [13].

RIPPER (Repeated Incremental Pruning to Produce Error Reduction) by Cohen [40] is a representative case which uses separate-and-conquer strategy and can run in ordered mode to extract a decision list. As the first rule learning technique that effectively handles the overfitting problem via incremental pruning and yet the state-of-the-art in inductive rule learning, RIPPER integrates a post-processing phase for optimizing a rule set by removing rules from previously learned rule set and relearning them in the context of both previous rules and subsequent rules [41] [13].

B. DECISION TREE

Despite sharing some similarities, decision trees differ from decision rules in terms of expressivity and learnability [42]. As mentioned above, whereas decision rules can be represented as ordered rule sets, decision trees are regarded as unordered rule sets. Decision trees are classification models whose structures consist of a number of nodes and arcs. Whereas a node is labeled by an attribute name, the from this node originated arcs are assigned a valid value associated with the attribute individually [8]. The hierarchical structure is beneficial for eliminating overlapping in rule sets, which makes classification easier. However, it can lead to more complex rules and make decision trees large and difficult to interpret [42] [43]. In general, decision rules/lists are more expressive than decision trees, and hence easier for humans to understand. In practice, decision trees are often converted into decision rules after being trained. Among many decision tree induction algorithms, the most popular ones are C4.5, developed by Quinlan [44], and its variants [8].

C. INDUCTIVE LOGIC PROGRAMMING

The term inductive logic programming (ILP) was defined by Muggleton [45] as the intersection of inductive learning and logic programming. By making use of logic programs as expressive representation for examples, background knowledge (BK) and hypotheses, ILP can learn complex relational theories and renders itself distinguishable from most other machine learning techniques. Given positive and negative examples along with BK, commonly in the form of Horn clauses, an ILP system aims at forming a hypothesis which explains examples in terms of BK and entails as many positive and as few negative examples as possible [45] [46]. With the use of BK as a form of inductive bias, ILP systems can normally generalize from small numbers of examples, which addresses one of the major limitations of current machine learning techniques, i.e., the need for a large amount of training data [47]. This distinct advantage of ILP can be further exploited by leveraging recursion, i.e., having the same predicate in rule body and head, which is regarded as one of several improvements of current ILP techniques [47] [48]. Like selecting appropriate features in common machine learning approaches, choosing appropriate BK is seen as a crucial step in learning process of ILP. To address the limitation that ILP has traditionally relied on manually designed BK by human experts, a shift to automatically learning BK is supported both by inventing new predicate symbols and by performing lifelong and transfer learning to discover reusable knowledge [47].

D. ASSOCIATION RULE LEARNING

As one of the most popular data mining methods, association rule learning is applied on a great variety of applications for discovering correlations among a set of attributes in a dataset and representing them as association rules [49]. Although the idea of association rules originates in 1960s with the aim of automatically generating numerous statistical hypotheses in the form of association rules [13], the arguably first association rule learning algorithm, i.e., Apriori algorithm, was designed by Agrawal and Srikant [50] in 1994 for frequent itemset mining and association rule learning over relational databases. A so-called breadth-first level-wise search is performed for acquiring all frequent itemsets, which are then converted into association rules in a post-processing phase [13].

The importance of an association rule is often measured with two factors, i.e., support and confidence. Whereas support denotes the proportion of all items in a database satisfying both rule body and head, confidence indicates the ratio of number of items satisfying both rule body and head to number of items satisfying rule body [50]. To overcome the problem that Apriori algorithm may suffer from large computational complexity for rule extraction in a dense database, a large number of successor algorithms with modifications mostly conducted on finding frequent itemsets are proposed for performance boost, e.g., [51] [52].

E. FUZZY LOGIC

When dealing with continuous attributes in rules, sharp boundary or strict threshold in attribute values can often cause classification system to make inaccurate or unfair decision. To solve this problem, fuzzy logic proposed by Zadeh [53] is normally applied to allow blurry boundary or threshold, which is enabled by discretizing continuous values into categories, typically represented as linguistic variables (e.g., high, middle and low). Meanwhile it introduces membership function like triangular function, sigmoid function or Gaussian function. A value between 0 and 1 is used for describing membership degree in each category, which hence eliminates a sharp boundary between categories. Also known as possibility theory, fuzzy set theory is seen as an alternative to traditional bivalent logic, i.e., either true or false. Whereas in traditional crispy sets every element only belongs to one set/category, in fuzzy set theory elements can belong to more than one fuzzy set with different membership degrees. Fuzzy logic allows a higher abstraction level and offers more flexibility when dealing with imprecise data [43].

A very successful application of fuzzy logic is the fuzzy controller firstly explored by Mamdani *et al.* [54], which is a reasoning system composed of a fuzzification module, a fuzzy rule base, an inference engine and a defuzzification module [3]. To investigate the possibility of human interaction with a learning controller, heuristic control rules stated by a human operator are transferred into an automatic control strategy in the fuzzy controller. The control strategy set up linguistically can prove to be very effective [54].

F. ROUGH SET THEORY

Although often contrasted with fuzzy set theory, rough set theory, introduced by Pawlak [55], is another independent approach to imperfect knowledge, and their relationship should be considered as complementary rather than competitive [55] [56]. Unlike fuzzy set theory which needs additional information about data, e.g., membership degree, rough set theory has the merit of not relying on any preliminary information about data [57]. As a mathematical tool to deal with vagueness and uncertainty from imprecise and insufficient knowledge, the main idea behind rough set theory is an indiscernibility relation associated with a set of attributes. A rough set is a formal approximation of a (original) crisp set, which results in a pair of crisp sets, called as lower approximation and upper approximation sets [56]. Whereas lower approximation set represents a lower boundary of the target set, upper approximation set represents an upper boundary of the target set. In any case, rules derived from lower approximation are certainly valid and rules extracted from upper approximation do not always hold true. Given real-world data, oftentimes some classes can not be distinguished only making use of a set of available attributes. Rules induced by employing rough set theory, which approximately defines such classes, are more general than information contained in the original imprecise or noisy dataset. Thus new data samples may be more correctly classified by these rules. An example is the data system LERS (Learning from Examples based on Rough Sets) presented by Grzymala-Busse [58], which induces rules from data with conflicting objects, i.e., data inconsistency. Conflicting objects appear if objects of different classes have the same values for all current existing attributes.

G. GENETIC ALGORITHM

Unlike the aforementioned rule induction approaches which use existing examples and background knowledge to generate their first set of rules, genetic algorithms (GA), as another family of stochastic separate-and-conquer rule learning algorithms for finding good rules [59] [8], perform a randomized global search in solution/hypothesis space and randomly generate a collection of solutions/hypotheses as the first set of rules. GA, introduced by Holland [60] as an inspiration of natural selection and evolution, have laid a foundation for a number of other techniques. Whereas a rule, also called as solution or hypothesis, is regarded as an individual, the newly derived rules form a generation and the entire rule set is referred as a population. Rules are encoded as chromosomes, i.e., strings of attributes, in which an attribute is a bit in a string. Assume that in a given training set every instance consists of two Boolean attributes, i.e., A1 and A2, and belongs to one of two possible classes, i.e., C1 and C2. The rule "IF A1 AND NOT A2 THEN C1" can be encoded as the bit string "100" [43].

Generated candidate rules are evaluated by a fitness function, which is normally a weighted function of rule accuracy, complexity and other performance metrics. In order to let the selected best candidates evolve and pass some of their characteristics to their offspring, they are modified by applying two genetic operators called crossover, i.e., randomly exchanging conditions between rules, and mutations, i.e., randomly inverting conditions in a rule. Whereas crossover is considered as deterministic operator passing best attributes of two parent rules to an offspring, mutation is a probabilistic operator which tries to find new useful attributes. Moreover, rule constraints are satisfied by either introducing penalties in the fitness function or directly encoding them in the rule data structures [3]. The iterative process of generating new rules continues until one of predefined stopping criteria is met. GA are easily parallelizable and are employed for various optimization problems along with their role in data mining as fitness evaluation function for other algorithms [43]. But a fundamental problem of GA is that they could repeatedly generate ineffective rules due to randomized global search.

H. GENETIC PROGRAMMING

As an extension of the GA, genetic programming (GP) introduced by Koza [61] employs a different solution encoding method with the aim of solving more complicated real-world problems in a variety of fields. Generally speaking, whereas genome structures in GA are fixed-length strings that encode candidate solutions, structures of genome are expressed in GP as syntax trees rather than strings. A tree consists of multiple nodes and links, in which nodes indicate execution instructions and links denote the arguments for each instruction. In order to construct complicated programs, programs can be composed of multiple components in more advanced forms of GP, i.e., a set of sub-trees/branches grouped together under a root node to form a tree with complex structure [62]. As a systematic method for letting computers to automatically solve a problem, GP initially starts from a high level statement of what needs to be done and attempts to produce a computer program to solve it. Certain well defined preparatory steps can be specified by humans. Then GP iteratively evolves a population of computer programs by applying genetic operators like crossover, mutation, reproduction, gene duplication, and gene deletion [62].

I. GENETIC NETWORK PROGRAMMING

To overcome a GP's fundamental problem "bloat", i.e., the search for better programs halts after certain generations as the programs become too large, a new type of evolutionary method named genetic network programming (GNP) is proposed by Hirasawa *et al.* [63]. With integration of a directed graph structure for its individual representation, GNP exhibits great expressiveness in modeling complicated programs/problems and hence can overcome the low searching efficiency of GP [63]. Originally, this graph-based evolutionary algorithm with network structure representing its genome is developed to leverage the more expressive representation ability of graphs, i.e., with reusability of nodes, for applications in dynamic environments. A directed graph in GNP contains two types of nodes, i.e., judgment node and processing node, which allow flexible representation and recombination of rule attributes. Another structural property of GNP beneficial for handling dynamic problems is that a previous node transition can affect the current node to be used, called as implicit memory function [64].

Moreover, the potential of GNP is further exploited in [64], in which GNP is coupled with reinforcement learning (RL) to create effective graph structures for producing better results. Besides, an unique graph-based feature of GNP "transition by necessity", i.e., only activating relevant nodes and the transitions for one particular task, is studied in [65] along with traditional genetic operators like crossover and mutation for evolving the directed graphs. The proposed simplified genetic operators alleviate unnecessary difficulty for evolution and can efficiently evolve even more compact programs [65].

J. LEARNING CLASSIFIER SYSTEMS

Yet another rule learning technique with great dependence on GA is known as learning classifier systems (LCS), which were introduced also by Holland in the work [60]. Although LCS are less famous than GA, with a raising number of application areas LCS are gaining more and more visibility in scientific research [66]. Two biological components, i.e., evolution and learning, are employed in LCS, in which the evolution process is guided by the learning process to induce better rules. The evolutionary component plays a key role in discovering novel rules and is embodied by a discovery mechanism often applying GA. The learning component is responsible for assigning credit to rules and is embodied by a learning mechanism often employing RL [67]. Both mechanisms rely on the system environment, i.e., limited source of input data. By interacting with this environment, LCS receive feedback in the form of numerical reward which drives the learning process [68].

Note that there are two forms of GA, namely generational GA and steady state GA. Whereas in generational GA a population is renewed from one generation to the next, in steady state GA individuals are renewed in the population one by one without notion of generation [66]. Analogically there are two distinct LCS styles, i.e., Michigan style by Holland and Pittsburgh style by Smith [69]. Whereas in Michigan style LCS the GA operate at the level of individual rules, the GA in Pittsburgh style LCS operate at the level of an entire rule set. Although Pittsburgh style LCS have the advantage of circumventing the potential problem caused by credit-sharing among individual rules, they suffer from heavy computational requirements in evolving multiple rule sets simultaneously. In contrast to Pittsburgh style, the Michigan style has drawn more attention and is seen as the standard LCS framework as it can be widely applied online in a broader range of problem domains [66] [68]. Another major criterion for LCS division is how reward in RL is taken into account in fitness function, which results in strength-based LCS [70], accuracy-based LCS [71], and anticipation-based LCS [72]. The XCS (eXtended Classifier System) [71] as a representative accuracy-based LCS is notably one of the most studied LCS [66].

K. PARTICLE SWARM OPTIMIZATION ALGORITHM

Like the aforementioned Pittsburgh style, another population-based heuristic rule learning strategy called particle swarm optimization (PSO) algorithm also targets at the level of an entire rule set for rule evolution, i.e., taking each rule set as an individual instead of a single rule in a rule set. PSO algorithm is inspired by social behavior through simulation of bird flocking and has already found a wide range of applications across different fields shortly after its introduction in [73] by Kennedy *et al.* [74]. Thanks to its ability to achieve a fast convergence, insensitivity to population size, and high scalability, PSO algorithm proves to be effective in optimizing complex multidimensional discontinuous problems in various research areas [75]. As a stochastic evolutionary algorithm based on swarm intelligence, PSO algorithm may outperform GA due to its simple implementation and hence reduced effort for optimizing hyperparameters, i.e., parameters of an algorithm. PSO is based on swarms of individuals called particles and every swarm is regarded as a solution in the solution space. The goal of PSO algorithm is to efficiently evolve a set of coadapted rules that cooperate with each other to solve a given problem [76]. Like in GA, a set of rules are randomly initialized and distributed in the multidimensional search space and a fitness function is employed to iteratively select and evolve rules by modifications. At every iteration, position and velocity of a particle are updated by means of three components, i.e., inertial component, self-recognition component and social component. In contrast to GA's three main operators, i.e., selection, crossover and mutation, PSO has just one major operator, namely velocity, which is represented as a matrix with the particle dimensions [76].

L. SEQUENTIAL PATTERN MINING

Whereas the rule induction techniques mentioned above mostly focus on stateless input data, i.e., data not in an order with timestamp, another rule learning technique introduced by Agrawal and Srikant [77] is known as sequential pattern mining (SPM), which is designed to discover symbolic sequences in data with a concrete notion of time. In SPM, a sequential pattern is a frequent subsequence in a sequence, or sometimes also called as a sequence in another larger sequence. A sequence is composed of several consecutive states/events ordered by timestamp, and a state/event can have a number of attributes, which can be represented as rule conditions as aforesaid. Besides, the sequential ordering of states is also taken into account in the rule bodies, which makes SPM track the operating states and thus be able to find extra details for rule construction.

Other rule learning techniques like association rule mining may fail to discover important useful patterns in some data by ignoring the sequential relationship between states, i.e., information about the "dynamics" in data. This holds true especially in some domains [78], like in energy consumption behavior recommendation [79] where data are naturally encoded as sequences, and in network intrusion detection where order of events is also important. A close analogy is stateful firewall[2] [80]. In data mining, two common forms of sequence data are time-series, i.e., an ordered list of numbers, and symbolic sequence, i.e., an ordered list of nominal data (symbols) [43].

M. EPISODE MINING

The methodology of SPM can be extended to mining periodic sequential patterns, partial order patterns, trees, lattices, episodes etc. by introducing user-specified constraints, folding events into proper-size windows or relaxing the requirement of strict sequential ordering [43]. An independently proposed technique called episode mining by Mannila *et al.* [81] may be referred to as constraint-based SPM with the aim of reducing the search space. As per Mannila *et al.* [81], an episode is defined as a small partially ordered set of events that frequently occur in the sequential data within a given time interval. Such small temporal patterns/episodes of interest can be discovered to construct rules for describing or predicting the sequence behaviors. Although episode mining shares many similarities with SPM, their major difference is that episode mining targets at patterns appearing in a single sequence instead of a set of sequences. Episode mining is applied in various domains such as web-click streams, telecommunication, network traffic, sensor readings [82] [78]. One should bear in mind that episode mining and sequential pattern mining along with other pattern mining approaches may sometimes be used interchangeably in the literature.

N. MARKOV CHAIN

Another rule learning technique relying on (temporal) sequence is Markov chain, which is a very important type of stochastic process for describing a number of possible events based on conditional probability. The usual representation of labeled Markov chain as transition system or automata is exploited to model normal system behaviors with respect to certain contexts, and the normal behaviors can then be formalized as a set of rules. A labeled Markov chain is a state transition graph with every state having a unique label, which can be used to define a probability distribution and the probabilities are assigned to the directed transition arcs, i.e., discrete time Markov chains [83]. In Markov chains, the conditional probability of choosing a next system state only depends on the current system state, which is known as Markov property [84]. Markov chains also lay the foundation for other Markov models, such as hidden Markov models in which every sequential system state is only partially observable. Hidden Markov models are applied in a wide range of fields to recover data sequences that are not immediately observable, especially in speech recognition [85]. Moreover, hidden Markov models are also applicable for inferring behavior rules [26].

O. BAYESIAN NETWORK

As per Han *et al.* [43], mining graphs and networks represents a more general class of structures than sequences and trees, and can find a wide range of applications. Bayesian networks, viewed as a form of probabilistic graph for knowledge representation and reasoning, can be learned from data and then further transferred into a set of probabilistic classification rules for even more compact and interpretable knowledge representation [86]. A Bayesian network is composed of a network structure, i.e., a directed acyclic graph, and a set of probability tables. While the nodes/vertices in the network represent variables, links/arcs denote dependence between the corresponding variables. Given the values of relevant attributes, the posterior probability distribution of the class node can be inferred via different inference algorithms.

[2]While a stateful firewall performs better at identifying attacks, a stateless firewall is normally faster and still suitable for heavy network traffic loads.

Bayesian networks are very popular in data mining due to their abilities to deal with incomplete dataset; to learn causal relationships; and to combine prior knowledge with patterns learned from data [87]. Bayesian networks can be either manually built by experts or automatically learned from data, in which data features are used to construct nodes. A hybrid way to create Bayesian networks [88] is to use expert knowledge to build only a network structure and leverage data to learn the conditional probability tables for the structure, in which the learning can be conducted using empirical conditional frequencies from data [89]. As per Hruschka *et al.* [86], the Markov blanket strategy[3] can be integrated to select only the most influential attributes to be used in rule antecedents while extracting rules from Bayesian networks, which hence can reduce the number of conditions in rules.

P. RADIAL BASIS FUNCTION NETWORK

Another case where rules are induced from network structure is radial basis function (RBF) network based rule extraction. Along with providing a good solution to many pattern recognition and classification problems, RBF networks as a local Gaussian representation technique enable an easy conversion of the hidden units into symbolic rules [91]. A RBF network, proposed by Broomhead and Lowe [92], is a type of feedforward artificial neural network, which uses a radial basis function as activation function in its hidden layer. There are typically only three layers in RBF networks, i.e., an input layer, a hidden layer and an output layer of linear units, hence also known as shallow neural network. The number of hidden units is directly related to the dataset complexity. Unlike multilayer perceptron (MLP) networks, RBF networks provide a local learning system [93] that contains elements responsive to only a part of the input space and is particularly helpful to rule extraction [91]. In general, as knowledge learned by neural networks is distributed across the internal parameters and mostly not well-interpretable for humans, extraction of meaningful rules is regarded as an important and powerful technique for neurosymbolic integration within hybrid systems [94]. The local nature of RBF networks provides a very useful mechanism that can interpret the input to output mappings of networks in the form of symbolic rules [91].

VI. DATASET AND FEATURE ENGINEERING

Reliable datasets and well-performed feature engineering are recognized as a highly crucial part for efficient rule learning, as they are in any machine learning and data mining approach. Data are seen as observations of real-world phenomena, in which a limited aspect of reality is obtained through a small window provided by each piece of data [95]. However, the collection of all these observations normally gives us a messy and noisy picture with missing pieces. Still, by means of appropriate feature engineering, these often incomplete, redundant and occasionally partially wrong data are of great interest and importance to be exploited in machine learning processes in order to acquire insights and make predictions. A feature or an attribute is a numeric representation of an aspect of raw data and is taken as input in machine learning models. Oftentimes the right features can only be earned in the context of both the model and the data, which makes it difficult to generalize the practice of feature engineering across projects [95]. In rule learning, there exist 3 types of features, i.e., simple features, time-stamped features and contextual features. Oftentimes, the contextual features are of critical importance for distinguishing abnormal behaviors from attacks.

[3]The concept Markov blanket, introduced by Pearl [90], is applied to extract useful features from all available ones. It states that the conditional probability distribution of a given node in a Bayesian network is only influenced by the closely located nodes.

A. DATASET FOR IDS

Unlike other machine learning application fields, e.g., computer vision and natural language processing, in which available datasets are seldom a problem, the lack of datasets exposes one of the most challenging hurdles of applying machine learning in intrusion detection. As it can be seen in image recognition and speech recognition research areas, publicly available datasets can speed up their developments greatly. Hence the dire need for appropriate datasets is very obvious, in order to accelerate progress in machine learning based IDS.

There are different kinds of IDS datasets, such as public dataset and private dataset according to its availability; real-world dataset and simulated/synthetic dataset in terms of data source; attack-inclusive and attack-free dataset by presence of attacks. Simulated datasets, though, can not be treated equally, since some are generated purely from computer simulation programs and others are collected from test bed simulation with real components, often deemed as more realistic. Analogically, attack-inclusive datasets are not of the same importance. While some datasets contain only a single type of attack, other datasets are composed of numerous different attack vectors. Whereas some attack scenarios are emulated online, other attack data are just offline synthesized. One should bear in mind that how carefully and realistically the attack data are generated is crucial for training an effective machine learning model and extracting meaningful rules in practice.

In machine learning based IDS research, more often than not, evaluation is conducted on the currently "deprecated" benchmark datasets, i.e., DARPA dataset and/or its successors KDD CUP 99 dataset and NSL-KDDCUP dataset, as it can be seen in the Section VII. However, the DARPA dataset is often criticized by researchers [96] [97] due to its unrealistic attacks, redundancy etc. A summarized comprehensive list of publicly available datasets can be found in the work [24]. However, to the best of our knowledge, no reliable dataset generated from (simulated) energy systems appears publicly available at the present time. We strongly believe that an effective IDS should be trained using domain specific data.

Another common dataset problem is the dominant presence of unbalanced datasets, in which observations in one class significantly outnumber observations in the other class(es). This again especially holds true in machine learning based IDS research, since reliable attack data are really seldom available. Unbalanced dataset can influence the learning performance in an unfavorable manner, because learning system may have difficulties to gain enough information related to the minority class(es). Nevertheless, a number of data sampling methods are proposed to combat the learning difficulties due to unbalanced dataset. Two very common ones are random over-sampling, i.e., random replication of observations in minority class , and random under-sampling, i.e., random elimination of observations in majority class. As per [98], the very simple random over-sampling proves to be more useful than under-sampling and very competitive to more complex over-sampling methods.

B. FEATURE ENGINEERING

Feature engineering, often known as a data (pre-)processing step, is the process to extract meaningful features from raw data with the goal of improving learning performance. Having the right features can alleviate unnecessary difficulties of modeling and hence make model yield better prediction results as well. However, machine learning practitioners agree that feature engineering is predominantly a time consuming task and can take the vast majority of time in building a machine learning pipeline [99] [95].

Feature engineering includes feature transformation, i.e., creating more discriminatory features from the existing ones; feature selection, i.e., removing unnecessary features by means of feature filtering, wrapping, embedding [95]; dimensionality reduction, i.e., transforming feature space into a lower dimension via numerous techniques like PCA (principal component analysis), autoencoder; and feature scaling, i.e., data standardization, normalization etc. An often used example of feature engineering is that categorical data, including nominal and ordinal variables, are often converted into numeric representation, since machine learning algorithms require numeric inputs. One common approach is one-hot encoding, in which each categorical type is mapped to a binary vector and thus it ensures higher numbers not assumed to be more important during learning. Since feature engineering can prove to be labor-intensive and sometimes tedious works, a number of research works endeavor to automate feature engineering, e.g., [99].

C. PERFORMANCE METRICS

As per [100], the purpose of evaluation is to estimate model performance; to determine the most suited model; and to convince potential users. That is to say, it aims at finding the model with *best performance*. The term *best performance* is actually hard to define and mostly context-dependent, since there are various performance metrics that have (slightly) different viewpoints about what is better. One could refer them as "competitor" metrics owing to existing trade-offs. There are a number of ways to select data from an original dataset for evaluation. The simplest one is arguably the hold-out validation, i.e., randomly choosing a portion of the dataset, along with two other also very popular methods, i.e., k-fold cross-validation and leave-one-out cross-validation.

Prediction performance is measured by comparing the predicted class and actual class, and then estimating how well the trained model can make predictions. One key concept in classification performance is the confusion matrix (Table 2), which calculates the frequencies of each possible prediction outcome and forms the basis for calculating many other performance metrics. The four possible prediction outcomes are: true positive (TP), true negative (TN), false positive (FP), false negative (FN). The mostly used performance metrics in IDS are: detection rate, also referred as true positive rate (TPR), sensitivity, recall; false alarm rate, also called false positive rate (FPR), fall-out; precision, also called positive predictive value; accuracy; misclassification rate; F_1 score, or F-measure. Most of these performance metrics are self-explanatory by means of formulas (1) to (5). As defined in formula (6), the F-measure is the harmonic mean of precision and recall. While recall indicates how complete the found positive instances are, precision denotes how often the positive predictions turn out to be correct [100]. Another widely used performance measure is ROC (receiver operating characteristic) index, i.e., ROC curve and AUC (area under the curve). ROC curve plots TPR against FPR, and shows TPR and FPR for various threshold values. It is often seen that ROC curves of various models are presented on a single ROC plot for easy performance comparison. AUC measures the area under a ROC curve and a higher AUC normally indicates better performance.

TABLE 2. Confusion matrix

		Actual class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

$$\text{Detection rate} = \frac{TP}{TP+FN} \tag{1}$$

$$\text{False alarm rate} = \frac{FP}{FP+TN} \tag{2}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{3}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{4}$$

$$\text{Misclassification rate} = \frac{FP+FN}{TP+TN+FP+FN} \tag{5}$$

$$F = 2 \cdot \frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}} \tag{6}$$

VII. RULE LEARNING BASED IDS

As mentioned in Section III, the major distinction between rule learning based IDS and other types of IDS is that in rule learning based IDS human interpretable rules are learned using ongoing training data and an intrusion detection engine can be periodically provided with updated rules. Mostly, these rules can either describe normal system behaviors or model normal protocol behaviors, i.e., system specification or protocol specification.

In the following, we study various application cases of rule learning based IDS which are categorized by the implemented techniques. Each type of IDS is presented mainly by only one or two selected articles, since this work primarily serves to provide an overview of how these rule learning techniques can be applied in intrusion detection research. Table 3 summarizes and compares the reviewed rule learning based IDS.

A. DECISION RULE BASED IDS

In [101], Helmer *et al.* study system call traces based IDS using the decision rule algorithm RIPPER for rule induction assisted with a genetic algorithm for feature selection. The presented multi-agent distributed IDS consists of data gathering agents that collect system logs and audit data, low level agents that monitor and classify ongoing activities, and data mining agents that use machine learning to discover predictive rules for intrusion detection. Inspired by the success of the bag-of-words representation of documents, they propose a feature vector representation to describe the system calls executed by privileged programs, in which a feature vector represents a process and each trace is composed of a few feature vectors. This feature vector approach contributes to the simplicity of the learned rules, which not only allows the application in near real time intrusion detection, but also eases the rule discovery process for learning algorithms and helps human expert examine rules.

B. DECISION TREE BASED IDS

Sindhu *et al.* [102] propose a so-called lightweight intrusion detection system, in which decision tree and neural network are combined to achieve better detection performance. In order to maximize the detection rate, a wrapper based feature selection algorithm is presented to identity suitable features after data pre-processing and removing redundant instances. A neurotree paradigm, which indicates the hybridization of improved neural network and decision tree with enhanced C4.5 algorithm, is proposed to overcome individual limitations and achieve synergetic effects for intrusion detection. With evaluation carried out on KDD CUP 99 dataset, the authors compared the proposed approach with six other popular decision tree classifiers, e.g., decision stump and random forest, and conclude that their approach can achieve the highest detection rate (98.4%) while keeping the error rate as the lowest (1.62%). Error rate is calculated as the sum of weighted false positive rate and false positive rate.

Similarly, Nancy *et al.* in [103] present an intrusion detection system which integrates decision tree with convolution neural networks. A feature selection algorithm called dynamic recursive feature selection algorithm (DRFSA) is proposed. The demonstrated system consists of a pre-processing module, a feature selection module, a decision tree classifier, a rule based decision manager, a fuzzy rule manager, a temporal constraints manager and a knowledge base. All the system modules are connected to the rule based decision manager which acts as coordinator in intrusion detection process. The authors demonstrate the performance of proposed system using KDD CUP 99 dataset and network traces recorded from the ns2 network simulator. The conducted experiments show that their approach can achieve not only better intrusion detection accuracy, but also higher packet delivery ratio and network throughput when comparing with other IDS using algorithms like enhanced C4.5 or SVM.

C. INDUCTIVE LOGIC PROGRAMMING BASED IDS

Ko [29] introduces a specification based IDS using inductive logic programming, which allows using complex security background knowledge in the learning process to generate reasonable and consistent detection rules. The author developed a specification induction engine by extending an existing ILP tool to automatically construct valid behavior rules of programs, irrespective of certain system vulnerabilities. The inductively learned specifications are easily understandable for humans and thus formally analyzable as well. Firstly, the Mode Directed Inverse Entailment [117] approach is

[4]Note that the performance of developed intrusion detection techniques in the listed references is often evaluated using only some specific attacks or only few attack samples, and thus can not be generalized for various attack scenarios or different target systems. In some cases, the performance, e.g., detection rate, is not directly given by the authors in numeric form, but can be concluded from the experimental results, like in [29]. Unfortunately, the false alarm rate is not always given in these references. Only providing detection rate does not make much sense, since one could always achieve 100% detection rate by simply classifying every observation as attack without even examining it. Similarly, misclassification rate or accuracy alone can not prove anything when dealing with highly unbalanced data. Any detection technique could obtain 0% misclassification rate or 100% accuracy effortlessly in an extreme case when testing data only contains benign samples.

[5]Not mentioned in the article specifically.

[6]Computer Immune Systems (CIS) Dataset from University of New Mexico https://www.cs.unm.edu/~immsec/systemcalls.htm

[7]Knowledge Discovery and Data Mining (KDD) Cup Dataset from University of California, Irvine http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[8]https://www.isi.edu/nsnam/ns/

[9]Defense Advanced Research Projects Agency (DARPA) Dataset from Lincoln laboratories at Massachusetts Institute of Technology, TCPDump files (network traffic) https://www.ll.mit.edu/r-d/datasets

[10]Network Security Laboratory (NSL)-KDD Dataset from University of New Brunswick https://www.unb.ca/cic/datasets/nsl.html

[11]GureKddcup Dataset http://www.sc.ehu.es/acwaldap/gureKddcup/

[12]Defense Advanced Research Projects Agency (DARPA) Dataset from Lincoln laboratories at Massachusetts Institute of Technology, BSM files (system call sequences) https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset

[13]Information Exploration Shootout (IES) Dataset from University of Massachusetts Lowell http://ivpr.cs.uml.edu/shootout/about.html

[14]The U.S. National CyberWatch Mid-Atlantic Collegiate Cyber Defense Competition (MACCDC) https://www.netresec.com/?page=MACCDC

TABLE 3. Summary of reviewed rule learning based IDS

References	Target system	Techniques	Protocols	Attack purposes	Attack types	Dataset	Features	Performance[4]
Helmer *et al.* 2002 [101]	Unix programs	RIPPER algorithm	SMTP	Privilege escalation	—[5]	CIS dataset [6]	System calls	Accuracy: 0.99 False positive rate: 0
Sindhu *et al.* 2012 [102]	Internet	Decision tree Neural network	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99[7]	Connection duration Network service etc.	Detection rate: 0.984 Error rate: 0.016
Nancy *et al.* 2020 [103]	Internet WSN	Decision tree Fuzzy logic	TCP/IP suite LEACH	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99 ns2 simulator[8]	Connection duration Network service etc.	Precision: 0.574-1 Recall: 0.283-0.98
Ko 2000 [29]	Unix programs	Inductive logic programming	TCP/IP suite	Privilege escalation	Buffer overflow	Simulated	System calls	Detection rate: 1
Stakhanova *et al.* 2007 [104]	MANET	Inductive logic programming	AODV DSR	Route disruption	fake request message fake response message	ns2 simulator	Network layer features	—
Li *et al.* 2010 [105]	Internet	Association rule learning	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	DARPA99[9]	Connection duration Network service etc.	Detection rate: 0.7 at False positive rate: 0.1
Elhag *et al.* 2019 [106]	Internet	Fuzzy association rule learning Evolutionary algorithm	TCP/IP suite	Denial-of-service Privilege escalation Probing	SYN flooding Port scanning Buffer overflow etc.	KDD CUP 99 NSL-KDDCUP[10] Gure-KDDCUP[11]	Connection duration Network service etc.	Detection rate: 0.78-0.96 False positive rate: 0.003-0.01 etc.
Nagarajan *et al.* 2011 [107]	Internet	Fuzzy logic	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99	Connection duration Network service etc.	Accuracy: 0.90-0.99 Precision: 0.05-0.99
Rawat *et al.* 2005 [108]	Unix programs	Rough set theory	IP suite	Privilege escalation	Buffer overflow etc.	BSM 98 [12]	System calls	Detection rate: 1 at False positive rate: 0.042
Patel *et al.* 2015 [5]	Internet	Genetic algorithm	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99	Connection duration Network service etc.	Detection rate: 0.987
Lu *et al.* 2004 [109]	Internet	Genetic programming	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	DARPA99	Connection duration Network service etc.	Detection rate: 1 at False positive rate: 0.014
Mabu *et al.* 2010 [49]	Internet	Fuzzy association rule mining Genetic network programming	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99 DARPA98	Connection duration Network service etc.	Detection rate: 0.987 at False positive rate: 0.005
Shafi *et al.* 2009 [110]	Internet	Learning classifier systems	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99	Connection duration Network service etc.	Overall accuracy: 0.92 at False positive rate: 0.006
Abadeh *et al.* 2006 [111]	Internet	PSO algorithm	TCP/IP suite	Denial-of-service Privilege escalation Probing	Smurf Ipsweep Buffer overflow etc.	KDD CUP 99	Connection duration Network service etc.	Detection rate: 0.942 at False positive rate: 0.011
Pan *et al.* 2015 [112]	Power transmission system	Sequential pattern mining	IEEE C37.118 Modbus TCP	Against distance protection	False data injection	Test bed simulation	Measurement data Various logs	avg. Detection rate: 0.73
Lee *et al.* 1998 [113]	Unix programs Internet	Association rule mining Episode mining	IP suite	Privilege escalation Probing	IP spoofing Port scanning etc.	CIS dataset IES dataset[13]	System calls Connection duration etc.	Misclassification rate: 0.009-0.42
Luo *et al.* 2000 [114]	Internet	Fuzzy association rule mining Fuzzy episode mining	IP suite	Privilege escalation Probing	IP spoofing Port scan	IES dataset	TCP ports TCP flags etc.	—
Ali *et al.* 2015 [37]	AMI	Markov chain Linear temporal logic	ANSI C12.22 PLC	Denial-of-service Probing	Scanning Mimicy attacks etc.	Real-world dataset Test bed simulation	Event logs	Accuracy: $>$ 0.95 at False positive rate: 0.0.001
Pan *et al.* 2015 [25]	Power transmission system	Bayesian network	IEEE C37.118 Telnet	Against over-current protection	False data injection Physical manipulation	Test bed simulation	Measurement data Various logs	Detection rate: 1
Ganesan *et al.* 2018 [115]	Internet	Bayesian abductive reasoning	IP suite	Privilege escalation Probing etc.	Port scanning etc.	MACCDC 2012[14]	Network layer features	—
Naik *et al.* 2018 [116]	Internet	Dynamic fuzzy rule interpolation	IP suite	Probing	Port scanning	Simulated	Packet frequency Time interval etc.	—

employed to confine and structure the search space of the suitable specifications. Then he specifies the evaluation criteria for the candidate solutions and a searching algorithm to find the best solution. To test the validity of proposed approach, an extended ILP tool is used to synthesize the valid specifications for more than 10 privileged programs in FreeBSD OS. However, the author discussed only the evaluation of generated specifications for two Unix programs against buffer overflow attacks very shortly. Whether all learned specifications are tested by implementing various attack vectors is not mentioned. Nevertheless, he concludes this work by saying that the automatically generated rules, regardless of possessing specific knowledge about the vulnerabilities, are very effective and accurate in detecting intrusions with a low false positive rate, and comparable with those rules manually developed by human experts. At the end, the author points out that the developed approach can also be applied to learn valid behavior of network protocols or services.

By extending previous work from unix programs to network protocols, Stakhanova *et al.* [104] present one of the first efforts to automatically generate specifications from observed run-time monitoring of routing protocols. Based on the assumption that specifications of mobile ad-hoc networks (MANET) routing protocols can be learned from the request-reply flow of network traffic using inductive logic programming, they aim at deriving abstract model of protocol behavior from background knowledge and individual observations. Each observation is represented by a sequence of routing messages initiated during a single route discovery. The authors present a generalization algorithm for inducing easily interpretable specifications in the form of graph for AODV (ad-hoc on-demand distance vector) and DSR (dynamic source routing) protocols. They use traces of valid protocol behavior from the network simulator ns2 to derive the specifications and validate them with implemented route disruption attacks via fake request and reply messages, respectively. The preliminary experiments show effective attack detection by observed violation against learned specifications. However, no systematic evaluation of developed approach against various attacks is conducted.

D. ASSOCIATION RULE LEARNING BASED IDS

Li *et al.* in [105] present an attack detection method based on association rule learning, which discovers user behavior patterns from network traffic data. The knowledge extracted from database can be applied as detection rules in IDS. They claim that the proposed Length-Decreasing Support constraint can improve the Apriori algorithm. The Length-Decreasing Support constraint is used to address some limitations of Apriori algorithm, which generates frequent patterns using constant support value, regardless of the length of discovered patterns. The proposed approach can prevent target system from exclusively generating a very large number of short patterns, as long but infrequent patterns are also desired. The experiment performed on the DARPA 1999 dataset shows that the Apriori algorithm with proposed Length-Decreasing Support is generally more efficient than the original Apriori algothrim, and can enhance the detection rate under the same false alarm rate. Unfortunately, the authors do not discuss the evaluation very much in detail.

In [106], Elhag *et al.* combine a multi-objective evolutionary algorithm and fuzzy association rule learning to extract a more compact and informative rule set in comparison with other similar rule learning algorithms. The so-called multi-objective evolutionary fuzzy system can be trained using different performance metrics as objectives to adapt to user's requirement in accordance with detection trade-offs, and hence is better suited for its individual applications. In order to build the fuzzy associative classifier, the first step is to mine fuzzy association rules from training dataset by employing a search tree, which lists all possible frequent fuzzy item sets. A pre-selection procedure is carried out to reduce the size of a rule set, which therefore makes it more interpretable. Then a genetic selection and tuning process with an evolutionary algorithm is introduced to acquire a more compact and accurate rule set. They evaluate the suggested approach with public datasets KDD CUP99, NSL-KDD and Gure-KDDCUP, respectively. The evaluation result shows that their approach can outperform several other similar rule learning techniques under various aspects.

E. FUZZY LOGIC BASED IDS

Shanmugavadivu *et al.* [107] propose a network intrusion detection system using rule learning technique based on fuzzy logic. They present a strategy for automatic generation of fuzzy rules. Note that fuzzy rules are often crafted manually by system experts. However, in case of having lots of input attributes, it is nearly unfeasible to generate fuzzy rules manually. The lowest and highest values in the range {min, max} of a feature of the training data are used as a criterion to select a subset of all available features, referred as discriminative or predictive features. Comparing the value range of a predictive feature in the normal data, e.g., {2, 5}, with the one in attack data, e.g., {4, 8}, simple rules can be generated like: rule 1 "IF predictive feature > 5, THEN it is attack"; rule 2 "IF predictive feature < 4, THEN it is normal." By replacing the numerical values with linguistic terms like high, medium and low using the triangular membership function, those rules can be transferred to fuzzy rules like: rule 1 "IF predictive feature is high, THEN it is attack"; rule 2 "IF predictive feature is low, THEN it is normal." The Mamdani fuzzy inference system [54] is utilized here to synthesize a set of fuzzy rules with respect to the given criteria: as few rules as possible; rule body as short as possible. The evaluation on a subset of KDD CUP 99 dataset shows that their approach has an accuracy above 90% for all types of attacks, but suffers from a very high false positive rate for privilege escalation attacks, i.e., R2L (remote to local) and U2R (user to root).

F. ROUGH SET THEORY BASED IDS

In [108], Rawat *et al.* present a rule discovery based IDS making use of rough set theory. Given that the boundary between normal behaviors and intrusive behaviors can often be blurry, the capability of rough set theory to deal with uncertainty and vagueness can eliminate ambiguity in decision-making process. Based on the assumption that attack behaviors in a host system have localized characteristics and thus can be reflected in a piece of system calls sequence, the author demonstrate that the application of rough set theory facilitates extracting rules to identify these attacks with high accuracy. Compact rough set rules for intrusion detection can be derived by discarding redundant attributes and expressed as simple IF-THEN rules rendering them suitable for online attack detection, easy interpretation and further analysis. A rough set based algorithm LEM2 [58], which follows a heuristic strategy for inducing rules, is implemented to discover the minimum set of detection rules, i.e., the smallest number of rules with sufficient coverage. The authors evaluate the developed method on DARPA 1998 BSM (basic security module) dataset, which contains audit logs collected from a Solaris host. They perform experiments with different sequence lengths of system calls, i.e., different rule lengths, and compare the detection rates and false positive rates. Detection performance improvement can be observed with increasing sequence length until it reaches 35. The worst false positive rate is 4.2% while keeping a 100% detection rate. Moreover, a slightly better detection result can be achieved by including a default rule that declares any previously unseen system call trace as intrusive.

G. GENETIC ALGORITHM BASED IDS

In [5], the authors propose a rule discovery based IDS using a genetic algorithm. The generated rules are represented as individuals in a population and every individual consists of encoded antecedent (IF part) and consequent (THEN part). The genome of an individual is represented by a sequence of n conditions which are formulated as a triplet *(predictor attribute, relative operator, value)*, e.g., *(Protocol_type, =, tcp)*, where n is the number of triplets. The consequent of an individual is the predicted class, which is not represented in the genome. In terms of hyperparameters of the implemented genetic algorithm for the training process, the authors choose tournament selection with tournament size equal to 5, three-point crossover with 95% probability of rule conditions swapped between individuals, 1% mutation probability and 2 elitists in each generation. With respect to confidence, coverage and comprehensibility of generated rules, a fitness function is constructed as rule evaluator for reproducing individuals/rules with better performance. The output of this function is calculated as the sum of weighted values of confidence, coverage and comprehensibility, and then normalized in the range 0 and 1. The experimental results show that the overall detection rate is 98.7% and there is a slight variation for different attack types. However, the authors neither mention the false positive rate of detection results nor discuss the learned rules further in detail, both of which are of critical importance with respect to practicality of proposed approach.

H. GENETIC PROGRAMMING BASED IDS

In [109], Lu *et al.* investigate a genetic programming based approach to extract rules for attack detection. Mostly based on the idea of GA, genetic programming, however, replaces chromosomes with more complex data structures, i.e., parse trees, to represent rules, which enables higher representation ability of derived rules. Such parse trees comprise internal nodes and leaf nodes, in which internal nodes are called as primitive functions, i.e., AND/OR operators of the antecedent in a rule, and leaf nodes are called as terminals, i.e., conditions in the antecedent. For the training process, four genetic operators are introduced, i.e., reproduction, crossover, mutation and condition-dropping operators. A fitness function is constructed with two weighted parameters, namely support and confidence. While support refers to the ratio of the number of observations covered by the rules to the total number of observations in the data, i.e., coverage of rules, confidence is represented by the ratio of the number of observations satisfying both antecedent and consequent to the number of observations only needing to match the antecedent, i.e., accuracy of rules. The authors evaluate proposed approach with DARPA 1999 dataset, which shows that the detection rate is almost 100% while the false positive rate is between 1.4% and 1.8%. However, to achieve 0% false positive rate, the detection rate can only reach 40%.

I. GENETIC NETWORK PROGRAMMING BASED IDS

By further exploring the aforementioned evolutionary algorithms, i.e., GA and GP, Mabu *et al.* [49] present a genetic network programming based technique for generating intrusion detection rules, which are extracted using directed graph structures rather than strings (used in GA) and trees (used in GP). Whereas in GA and GP a rule is directly represented as an individual in a generation, in GNP an individual itself is not a rule but a directed graph, which can generate several rules. This directed graph is built with three types of nodes, i.e., start node, judgment node and processing node, whose reusability renders more compact structure for deriving rules. While judgment nodes are identical with conditions, whose connections form the antecedent of a rule, processing nodes serve as action function which enables extracting various rules from a single graph. Like in GA and GP, selection, crossover and mutation are implemented as genetic operators in GNP. In this work, two fitness functions are designed, of which the first one is for evaluating quality of derived rules and the second one is used to assess ability of individuals at generating accurate rules. Moreover, the main objective of the proposed approach is to derive as many as accurate rules instead of creating optimal individuals, which, though, are not necessarily contradictory.

To identify intrusions using rules, the authors leverage both “misuse” detection and “anomaly” detection. While in “anomaly” detection they only form “benign” rule set using normal data, a “malicious” rule set is constructed with attack data in “misuse” detection to assist “benign” rule set for detecting attacks. While in “anomaly” detection the objective is to find as many normal rules as possible and explore the normal behavior space, they aim at mining more accurate rules in “misuse” detection, i.e., favoring quality over quantity. The authors evaluate the proposed “misuse” detection approach with KDD CUP 99 dataset and “anomaly” detection approach with DARPA 1998 dateset. Experimental results show a higher detection rate with a tolerable false positive rate in comparison with many other machine learning techniques. Furthermore, they investigate the detection performance improvement due to integration of fuzzy set in developed approach. When dealing with continuous attributes, fuzzy set can overcome sharp boundary problem, which enhances flexibility of mining more accurate rules.

J. LEARNING CLASSIFIER SYSTEMS BASED IDS

Shafi *et al.* [110] design a framework called UCSSE (sUpervised learning Classifier System with real time Signature Extraction) for automatically generating intrusion detection rules by applying learning classifier systems. Generalization and control mechanisms are developed in UCSSE to minimize overlap and conflict among detection rules by modifying rule boundaries, and to combat noisy and imbalanced data, respectively. As another genetic-based machine learning technique, LCS exploit the implicit parallelism of GA for dynamically and incrementally inducing rules for network intrusion detection with adaptability to environmental changes. A Michigan style LCS is employed in this work, which considers every individual rule as a classifier. A common problem in rule induction process is that a large number of redundant rules could be generated, which in turn negatively influences comprehensibility of the rule set and its applicability in time critical attack detection application owing to processing time. To overcome this problem, rule set pruning is proposed for finding a subset of derived rule collection which can achieve nearly the same coverage. Several modifications to the original UCS (supervised learning classifier system) are introduced by the authors to improve performance of developed systems, including a distance metric based function for classification, a biased class-distributive accuracy function, online GA rate adaptation, and fitness sharing technique in GA for promoting diversity in a population. The proposed method is evaluated on KDD Cup 99 dataset to demonstrate its detection performance. Significantly better accuracy, lower false positive rate, lesser amount of rules and more stable outcome can be achieved with UCSSE in comparison with original UCS.

K. PARTICLE SWARM OPTIMIZATION ALGORITHM BASED IDS

In this article [111], Abadeh *et al.* investigate a particle swarm optimization algorithm based procedure for automatically generating and adjusting fuzzy rules for intrusion detection. A rule is represented by an individual in a global population that is divided into some subpopulations representing different classes of rule head. This PSO algorithm based approach does not create new rules from "parent" rules, which makes it different from GA based approaches. The concept RAMS (Rule Antecedent Modification Sequence), which consists of several RAM (Rule Antecedent Modification) operators, is introduced for the heuristic local search procedure. This procedure aims to improve the quality of derived fuzzy rules by searching their neighborhoods with respect to some restrictions. As hyperparameters of the implemented PSO algorithm for the training process, the authors choose population size equal to 20, number of particles equal to 20, age weight parameter equal to 10000, 95% crossover probability, 10% mutation probability, and 20% replacement probability. Experiments carried out using the KDD CUP 99 dataset show that they can achieve an overall detection rate of 94.15% with a false positive rate of 1.1%. Besides, variation in detection performance is observed for different attack types.

L. SEQUENTIAL PATTERN MINING BASED IDS

Pan *et al.* in [112] introduce a sequential pattern mining based approach called common path mining for attack detection in power transmission systems. System states are represented by aggregating synchrophasor measurement data with power system audit logs and time stamps. The system state transitions are used to learn sequential behavior patterns or common paths. The common paths then describe expected normal behaviors as a sequence of system states and thus are referred to as specifications. A frequent-pattern (FP) growth algorithm [43] is applied in the training process to infer the common paths, i.e., the most frequently occurred system state sequences in a scenario, and dependent relationships between events. The authors introduce a number of different types of scenarios, i.e., various normal, faulty and attack scenarios. The proposed multiclass classification is performed by comparing observed system states to mined unique common paths of each scenario, and provides additional information in the detection result, such as which attack type the system is undergoing, and thus enables quicker incident response accordingly.

With respect to the fact that in power systems an anomalous or faulty system state can also be linked with actions unrelated to security, i.e., triggered by natural causes rather than attacks, the proposed approach can distinguish system faults and attacks based on the assumption that the system state transitions of all devices are not completely same in the case of system fault and in the case of under attack. That is to say, a single difference in system state transitions of all relevant devices can contribute to the distinction between anomalous behaviors due to natural causes and due to attacks. The presented IDS is evaluated using test bed simulation data, which gives an average 73.43% detection rate for previously unseen attacks. The authors point out that the common paths mining based IDS outperforms several other machine learning algorithms using the same dataset and is more suitable for high volume data stream in power systems.

M. EPISODE MINING BASED IDS

Both association rule mining and episode mining are used for intrusion detection in [113] [114]. Lee *et al.* [113] present an association rule learning algorithm and an episode mining algorithm for describing programs or user behaviors by computing the intra- and inter-audit record patterns, respectively. A systemic framework with agent-based architecture for intrusion detection is introduced to enable both efficient learning and real time detection, in which the learning agents continuously provide the detection agents with updated rules. The effectiveness in detecting intrusions gets improved by combining evidences from multiple so-called base classifiers that model diverse aspects of the target system behavior. They claim that promising results are demonstrated in their preliminary experiments both on host based intrusion detection using collected system call traces and on network-based intrusion detection using captured network packets.

In [114], the authors extend previous work [113] by proposing the integration of fuzzy logic into association rule mining and episode mining to produce more abstract and flexible rules for attack detection. It is based on the assumption that there are many quantitative features for intrusion detection and security itself is fuzzy. Taking advantage of fuzzy logic, more general rules for temporal statistical measurements can be produced at a higher and more abstract level than the data level, which enables detection of malicious activities even with certain variation. The experimental results of a proposed similarity evaluation function show a low similarity score between benign data and malicious data based on fuzzy association rules and a very low similarity score based on fuzzy episode rules.

N. MARKOV CHAIN BASED IDS

As one of the first works that leverage AMI configuration for deriving a Markov chain model representing normal system behavior, Ali *et al.* in [37] propose a robust mutation-based IDS that accurately models observed quasi-deterministic and predictable AMI behavior for intrusion detection while making it yet unpredictable for attackers. Given the fact that event log entries stored at metering data collectors exhibit a certain level of temporal dependence, a large sequence of log entries are utilized to learn a stochastic model based on Markov chain depicting AMI behavior. The specifications are written in Linear Temporal Logic (LTL). A sliding-window approach is employed for continuously learning model online. The features of a log entry include time stamp, source, forwarding and destination nodes, size and type of communication. To find the proper order of Markov chain, conditional entropy on different Markov chain orders are calculated and compared. The fourth-order Markov chain, in which a state consists of four consecutive log entries, is selected as it gives enough information to predict future states while keeping the model complexity still reasonable. Moreover, the authors design a configuration randomization module, i.e., mutating AMI behavior using secret key, to provide IDS robustness against evasion attacks.

The detection module is implemented in metering data collectors where it monitors both communication between smart meters and data collectors and communication between data collectors and metering head-end systems. In this way it potentially eliminates significantly high cost as deploying detection module in every smart meter. To validate the proposed approach, they use a real-world dataset gathered from thousands of smart meters of a utility provider, and synthetic data generated from some test bed simulations, which include different attack scenarios like DoS attacks, scanning, evasion and false data injection attacks. It is worth noting that the developed IDS aims only at large-scale attacks which result in destabilization of infrastructure, like compromising a large number of AMI devices simultaneously to cause a blackout. Meter tampering, i.e., energy theft by individual users, though, can not be caught, which is reflected in that metering values are not included as features in log entries for learning the Markov chain model.

O. BAYESIAN NETWORK BASED IDS

Pan *et al.* in [25] present an IDS for power transmission systems using a Bayesian network which models interdependencies between variables and graphically represents their casual relations. They design a test bed to simulate the studied power transmission system, which addresses attacks against over-current protection through different means including false data injection and sabotaging physical devices. The detection rules are derived from the constructed Bayesian network which consists of a number of paths. Every path can be considered as a distinct rule. The detection decision is made by comparing the sequence of so-called signatures. A signature includes a system state, its start time, actions, events and temporal distance to the previous signature, of which each combination of action and event is represented as the label of a vertex in the graph.

Although the Bayesian network developed in this work is constructed using human expert knowledge, we believe that this Bayesian network could be learned from data using appropriate algorithms. A common approach is to introduce an objective function that evaluates each network structure using training data, and then to search for the best Bayesian network according to this objective function [118]. Moreover, this work only studies a power transmission system with specific and simplified setting, which means that the developed IDS is not suitable for a different kind of system setting. In order to deploy such a Bayesian network based IDS in various system settings, an individual Bayesian network must be constructed for every single one of them, which can be obviously very burdensome. Besides, whereas in a simple case a Bayesian network can usually be constructed manually by human expert, it can be very hard for humans to specify the corresponding Bayesian network, when the target system is too complex.

P. RADIAL BASIS FUNCTION NETWORK BASED IDS

Although radial basis function networks have been being applied in various intrusion detection research works, the main focus remains on anomaly detection, such as [119] [120], instead of extracting intrusion detection rules. During our research, we could not find any article that utilizes radial basis function network to extract rules for intrusion detection. Nevertheless, radial basis function networks are exploited for rule induction in a few studies, e.g., [91] [121], irrespective of application areas. We believe that the developed techniques can be applied in rule learning based IDS, as per [119], which indicates another research opportunity. In [121], Jin *et al.* introduce a method for extracting interpretable fuzzy rules from RBF networks using regularization techniques. In order to extract interpretable symbolic rules from a RBF network and gain a deeper insight into the logical structure of studied system, they propose an adaptive weight sharing algorithm while keeping the number of basis in the RBF network small. An evaluation is carried out, in which 27 fuzzy rules are obtained from a RBF network with 27 hidden nodes using collected simulation data of an industrial process.

Q. RULE INTERPOLATION BASED IDS

Loosely speaking, rule interpolation based IDS can be regarded as rule learning based IDS. Here we distinguish rule learning based IDS and rule interpolation based IDS by the fact that the former ones generate rules mainly from data/observations without pre-existed rules and the latter ones derive rules based on existing rules. Nonetheless, both techniques are capable of detecting zero-day attacks leveraging newly induced rules, which make them outperform traditional misuse based IDS.

Based on the experience that new attacks often prove to be modifications of existing ones, Ganesan *et al.* in [115] propose a probabilistic abductive reasoning approach that leverages existing snort[15] rules tailored for known attacks to derive new rules for detecting evolved previously unseen attacks. A Bayesian model is trained on initial snort rules by identifying the correlation between attributes in rules, and is then used for abducing a number of new rules by replacing a set of attributes. In this way, the newly derived rules are capable of catching evolved attacks that are slightly different from known attacks.

In [116], Naik *et al.* introduce a dynamic fuzzy rule interpolation (D-FRI) based approach for intrusion detection, which exploits interpolated rules in order to improve the overall system coverage and efficiency. A transformation based FRI system is utilized with an initial sparse rule base to perform rule interpolation in order to generate a large amount of interpolated rules. The antecedents of newly generated

[15]Snort is a popular misuse detection engine for network intrusion detection. More information in: https://www.snort.org/

rules are allotted to a number of subgroups, of which the nonempty ones are selected as input to a GA-based clustering algorithm for finding a set of strong subgroups that contain many rules. The optimally clustered rules are then chosen for the subsequent rule promotion process. The authors integrate D-FRI with the IDS engine snort to present an intelligent and dynamic IDS which selects, combines, and generalizes informative interpolated rules, and merges them with the existing rule base. The experimental results show that the developed method can outperform standard snort due to enlarged coverage of dynamically learned rule base.

VIII. CHALLENGES AND PROSPECTS

The authors in [97] criticize the current machine learning based intrusion detection research for focusing mostly on enhancing detection rate while treating the systems as black boxes. They argue that a deep insight into systems' capabilities and limitations should be achieved for an effective deployment of such intrusion detection systems. As per [97], one can always find a machine learning technique for IDS that has slightly better performance in some specific training and evaluation setting. We emphasize that the statements hold especially true, when machine learning based IDS are implemented in critical infrastructures like smart grids, since the process of detecting intrusion should be made transparent and clear to human experts and security analysts for further investigation. There is a dire need to look for new domain specific approaches for machine learning based IDS with interpretable decision-making process.

Many in academia well-studied "best" detection techniques prove to be very effective with respect to a set of selected examples, however, may suffer in real life, because the intrusion examples can hardly be comprehensive. By incorporating extensive background security knowledge for learning algorithms, good specifications can be inductively generated using rule learning based techniques for discerning attacks from legitimate network traffics. These specifications can be of high quality not only regarding to examples but also in practice [29]. Specification based IDS present potentially the most effective attack detection techniques and meanwhile allow for lightweight intrusion detection to be implemented in systems with severe resource constraints [22]. A noticeable drawback of these approaches is that these specification rules ought to be redefined and updated continuously in an environment like SG in which there exist multiple ongoing alterations and modifications [23]. However, in the light of rule learning techniques, these efforts could be significantly reduced, which makes specification based IDS using rule learning appear to be very promising.

Nevertheless, given the fact that no single intrusion detection technique is capable of catching all attacks and/or producing zero false alarm, different detection approaches should be regarded as being complementary rather than competitive.

IX. CONCLUSION AND FUTURE WORK

In the present work, we provide a gentle introduction and a systematic analysis of intrusion detection systems, smart grids and rule learning techniques, respectively. Besides, we summarize the most important criteria for assessing quality of learned intrusion detection rules. Furthermore, to the best of our knowledge, we present the first survey focusing on rule learning based IDS to shed more light on this research area. We advocate that, when seeking opportunities to apply rule learning techniques in IDS, one should always bear in mind that no work can by no means include all applicable machine learning and data mining techniques for rule extraction. Hence it is also very important to explore new machine learning and data mining techniques for rule induction, and to make use of rule learning techniques already applied in other fields but not yet in IDS, along with trying to apply and improve the techniques mentioned in our article. One possible option would be to explore the potentials of various artificial neural networks for rule induction. As recently research topics move towards interpretable artificial intelligence, more advances are being made in this field, which may be further exploited for inferring knowledge in the form of symbolic rules. If viewed from a different angle, rule induction from artificial neural networks itself is a way to realize interpretable artificial intelligence.

It is worth noting that specifications or rules can be constructed or learned through different data sources, and represented in different abstract levels to detect or only to be able to detect a subset of all possible attacks. For instance, in [29] specifications are learned leveraging system calls to detect attacks against privileged programs, in [104] specifications are constructed only by means of network layer data which are more abstract than taking into account application data unit as well.

As mentioned in Section IV, there is no publicly available reliable IDS dataset for smart grids. Our future works include building real-world test beds for various subsystems in SG, especially micrograds due to not being commonly targeted in the research community [23], and AMI owing to its very critical importance for successful SG deployment. Since real attack data are not available and conducting real attacks in energy systems to collect data would be totally off limits, the best viable solution is to generate and collect normal and attack data on some carefully constructed test beds.

ABBREVIATIONS

AMI	Advanced Metering Infrastructure
AODV	Ad-hoc On-demand Distance Vector
AUC	Area Under the Curve
BAN	Business Area Network
BK	Background Knowledge
BSM	Basic Security Module
CPS	Cyber-Physical System
D-FRI	Dynamic Fuzzy Rule Interpolation
DARPA	Defense Advanced Research Projects Agency
DDoS	Distributed Denial-of-Service

DNP3	Distributed Network Protocol 3
DRFSA	Dynamic Recursive Feature Selection Algorithm
DSR	Dynamic Source Routing
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GA	Genetic Algorithms
GNP	Genetic Network Programming
GOOSE	Generic Object Oriented Substation Events
GP	Genetic Programming
GSE	Generic Substation Events
GSSE	Generic Substation State Events
HAN	Home Area Network
HMI	Human-Machine Interface
IAN	Industry Area Network
ICCP	Intercontrol Center Communications Protocol
ICT	Information and Communications Technologies
IDS	Intrusion Detection System
ILP	Inductive Logic Programming
IoT	Internet of Things
KDD	Knowledge Discovery in Databases
LAN	Local Area Network
LCS	Learning Classifier System
LERS	Learning from Examples based on Rough Sets
LTL	Linear Temporal Logic
MANET	Mobile Ad-hoc Network
MLP	Multilayer Perceptron
NAN	Neighborhood Area Network
NGFW	Next Generation Firewall
OPC	Open Process Communication
OSI	Open Systems Interconnection
PCA	Principal Component Analysis
PDC	Phasor Data Concentrator
PLC	Power Line Communication
PLC	Programmable Logic Controller
PMU	Phasor Measurement Unit
PSO	Particle Swarm Optimization
R2L	Remote to Local
RAM	Rule Antecedent Modification
RAMS	Rule Antecedent Modification Sequence
RBF	Radial Basis Function
RIPPER	Repeated Incremental Pruning to Produce Error Reduction
RL	Reinforcement Learning
ROC	Receiver Operating Characteristic
SCADA	Supervisory Control and Data Acquisition
SG	Smart Grid
SPAN	Switch Port Analyzer
SPM	Sequential Pattern Mining
TAP	Test Access Point
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
TPR	True Positive Rate
U2R	User to Root
UCS	sUpervised learning Classifier System
UCSSE	sUpervised learning Classifier System with real time Signature Extraction
VPN	Virtual Private Network
WAN	Wide Area Network
XCS	eXtended Classifier System

REFERENCES

[1] Homeland Security. *Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies*. 2016.

[2] E. D. Knapp. *Industrial network security: Securing critical infrastructure networks for smart grid, scada, and other industrial control systems*. 2nd edition. Waltham MA: Elsevier, 2014.

[3] P. P. Bonissone. "Soft computing the convergence of emerging reasoning technologies". In: *Soft Computing* (1997).

[4] M. Moradi and M. Zulkernine. "A Neural Network Based System for Intrusion Detection and Classification of Attacks". In: *Proceedings of IEEE International Conference on Advances in Intelligent Systems - Theory and Applications*. 2004.

[5] K. Patel and B. Buddhadev. "Predictive Rule Discovery for Network Intrusion Detection". In: *Intelligent Distributed Computing*. Ed. by Rajkumar Buyya and Sabu M. Thampi. Vol. 321. Cham: Springer, 2015, pp. 287–298.

[6] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou. "Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions". In: *Proceedings of the 9th ACM Conference on Computer and Communications Security*. 2002.

[7] C. Steven, D. Bruno, F. Martin, L. Ulf, S. Keith, and V. Alfonso. "Using Model-based Intrusion Detection for SCADA Networks". In: *Proceedings of the SCADA Security Scientific Symposium*. 2003.

[8] J. Fürnkranz, D. Gamberger, and N. Lavrač. *Foundations of Rule Learning*. Berlin, Heidelberg: Springer, 2012.

[9] W. Tong, L. Lu, Z. Li, J. Lin, and X. Jin. "A Survey on Intrusion Detection System for Advanced Metering Infrastructure". In: *Proceedings of the Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*. IEEE, 2016, pp. 33–37.

[10] R. Mitchell and I. Chen. "Behavior-Rule Based Intrusion Detection Systems for Safety Critical Smart Grid Applications". In: *IEEE Transactions on Smart Grid* 4.3 (2013), pp. 1254–1263.

[11] T. Miller. "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial Intelligence* 267 (2019), pp. 1–38.

[12] R. S. S. Kumar, A. Wicker, and M. Swann. "Practical Machine Learning for Cloud Intrusion Detection". In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 2017, pp. 81–90.

[13] J. Fürnkranz and T. Kliegr. "A Brief Overview of Rule Learning". In: *Proceedings of the 9th International RuleML Symposium*. Vol. 9202. Cham: Springer, 2015, pp. 54–69.

[14] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter and L. Kagal. "Explaining Explanations An Overview of Interpretability of Machine Learning". In: *Proceedings of IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE Computer Society, 2018.

[15] A. A. Aburomman and M. B. I. Reaz. "A survey of intrusion detection systems based on ensemble and hybrid classifiers". In: *Computers & Security* 65 (2017), pp. 135–152.

[16] C. V. Zhou, C. Leckie, and S. Karunasekera. "A survey of coordinated attacks and collaborative intrusion detection". In: *Computers & Security* 29.1 (2010), pp. 124–140.

[17] J. Arshad, M. A. Azad, R. Amad, K. Salah, M. Alazab, and R. Iqbal. "A Review of Performance, Energy and Privacy of Intrusion Detection Systems for IoT". In: *Electronics* 9.4 (2020), p. 629.

[18] D. Berman, A. Buczak, J. Chavis, and C. Corbett. "A Survey of Deep Learning Methods for Cyber Security". In: *Information* 10.4 (2019), p. 122.

[19] R. A. Bridges, T. R. Glass-Vanderlan, M. D. Iannacone, M. S. Vincent, and Q. Chen. "A Survey of Intrusion Detection Systems Leveraging Host Data". In: *ACM Computing Surveys* 52.6 (2020), pp. 1–35.

[20] A. L. Buczak and E. Guven. "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection". In: *IEEE Communications Surveys & Tutorials* 18.2 (2016), pp. 1153–1176.

[21] V. Chandola, A. Banerjee, and V. Kumar. "Anomaly Detection: A Survey". In: *ACM Computing Surveys* 41.3 (2009), pp. 1–58.

[22] R. Mitchell and I.-R. Chen. "A survey of intrusion detection techniques for cyber-physical systems". In: *ACM Computing Surveys* 46.4 (2014), pp. 1–29.

[23] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis. "Securing the Smart Grid: A Comprehensive Compilation of Intrusion Detection and Prevention Systems". In: *IEEE Access* 7 (2019), pp. 46595–46620.

[24] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho. "A Survey of Network-based Intrusion Detection Data Sets". In: *Computers & Security* 86 (2019), pp. 147–167.

[25] S. Pan, T. Morris, and U. Adhikari. "A Specification-based Intrusion Detection Framework for Cyber-physical Environment in Electric Power System". In: *International Journal of Network Security* 17.2 (2015).

[26] Roberto Di Pietro and Luigi V. Mancini, eds. *Intrusion Detection Systems*. Vol. 37. Advances in information security. New York: Springer, 2008.

[27] D. E. Denning. "An Intrusion-Detection Model". In: *IEEE Transactions on Software Engineering* SE-13.2 (1987), pp. 222–232.

[28] C. Ko, M. Ruschitzka, and K. Levitt. "Execution monitoring of security-critical programs in distributed systems: a specification-based approach". In: *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 1997, pp. 175–187.

[29] C. Ko. "Logic Induction of Valid Behavior Specifications for Intrusion Detection". In: *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 2000.

[30] G. Kim, S. Lee, and S. Kim. "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection". In: *Expert Systems with Applications* 41.4 (2014), pp. 1690–1700.

[31] E. D. Knapp and R. Samani. *Applied cyber security and the smart grid: Implementing security controls into the modern power infrastructure*. Amsterdam: Elsevier Syngress, 2013.

[32] E. Kabalci and Y. Kabalci. *Smart Grids and Their Communication Systems*. Singapore: Springer Singapore, 2019.

[33] M. Emmanuel, W. K.G. Seah, and R. Rayudu. "Communication Architecture for Smart Grid Applications". In: *Proceedings of IEEE Symposium on Computers and Communications (ISCC)*. 2018.

[34] K. Siozios, D. Anagnostos, D. Soudris, and E. Kosmatopoulos. *IoT for Smart Grids: Design Challenges and Paradigms*. Cham: Springer International Publishing, 2019.

[35] K. E. Martin, D. Hamai, M. G. Adamiak, S. Anderson, M. Begovic, G. Benmouyal, G. Brunello, J. Burger, J. Y. Cai, B. Dickerson, V. Gharpure, B. Kennedy, D. Karlsson, A. G. Phadke, J. Salj, V. Skendzic, J. Sperr, Y. Song, C. Huntley, B. Kasztenny, and E. Price. "Exploring the IEEE Standard C37.118–2005 Synchrophasors for Power Systems". In: *IEEE Transactions on Power Delivery* 23.4 (2008), pp. 1805–1811.

[36] IEC Technical Specification 62351–6. *Security for IEC 61850 profiles*. 2007.

[37] M. Q. Ali and E. Al-Shaer. "Randomization-Based Intrusion Detection System for Advanced Metering Infrastructure". In: *ACM Transactions on Information and System Security* 18.2 (2015), pp. 1–30.

[38] Peter Jackson. *Introduction to Expert Systems*. 3rd. USA: Addison-Wesley Longman Publishing Co., 1998.

[39] P. R. Rijnbeek and J. A. Kors. "Finding a short and accurate decision rule in disjunctive normal form by exhaustive search". In: *Machine Learning* 80.1 (2010), pp. 33–62.

[40] W. W. Cohen. "Fast Effective Rule Induction". In: *Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995.

[41] J. Fürnkranz. "Pruning Algorithms for Rule Learning". In: *Machine Learning* 27 (1997), pp. 139–172.

[42] J. Fürnkranz. "Decision Lists and Decision Trees". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, 2017, pp. 328–329.

[43] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. 3rd Edition. Morgan Kaufmann, 2012.

[44] J. R. Quinlan. *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[45] S. Muggleton. "Inductive logic programming". In: *New Generation Computing* 8 (1991).

[46] S. Muggleton. "Inductive Logic Programming: Issues, results and the challenge of Learning Language in Logic". In: *Artificial Intelligence* 114 (1999).

[47] A. Cropper, S. Dumancic, and S. H. Muggleton. "New Ideas in Inductive Logic Programming". In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. 2020.

[48] A. Cropper, A. Tamaddoni-Nezhad, and S. H. Muggleton. "Meta-Interpretive Learning of Data Transformation Programs". In: *Proceedings of International Conference on Inductive Logic Programming*. Cham: Springer, 2016.

[49] S. Mabu, C. Chen, N. Lu, K. Shimada, and K. Hirasawa. "An Intrusion-Detection Model Based on Fuzzy Class-Association-Rule Mining Using Genetic Network Programming". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41.1 (2010), pp. 130–139.

[50] R. S. R. Agrawal. "Fast Algorithms for Mining Association Rules". In: *Proceedings of the 20th VLDB Conference*. Morgan Kaufmann, 1994.

[51] J. Han, J. Pei, Y. Yin, R. Mao. "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach". In: *Data Mining and Knowledge Discovery* 8 (2004).

[52] B. Negrevergne, A. Termier, M.-C. Rousset, and J.-F. Méhaut. "Para Miner: a generic pattern mining algorithm for multi-core architectures". In: *Data Mining and Knowledge Discovery* 28.3 (2014), pp. 593–633.

[53] L. A. Zadeh. "Fuzzy Sets". In: *Information and control* 8 (1965).

[54] E. H. Mamdani and S. Assilian. "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller". In: *Man-Machine Studies* 7 (1975).

[55] Z. Pawlak. *Rough sets: Theoretical aspects of reasoning about data*. Dordrecht: Kluwer Academic Publishers, 1991.

[56] Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko. "Rough sets". In: *Communications of the ACM* 38.11 (1995), pp. 88–95.

[57] J. W. Grzymala-busse. "Knowledge acquisition under uncertainty - a rough set approach". In: *Intelligent and Robotic Systems* 1 (1988).

[58] J. W. Grzymala-Busse. "Rule Induction". In: *Data Mining and Knowledge Discovery Handbook*. Vol. 1. Boston, MA: Springer, 2005, pp. 249–265.

[59] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Reading, MA. and Wokingham: Addison-Wesley, 1989.

[60] J. H. Holland. *Adaptation in Natural and Artificial Systems*. Vol. 18. Cambridge, MA: The MIT Press, 1992.

[61] J. R. Koza. *Genetic programming: On the programming of computers by means of natural selection*. Cambridge, MA and London: The MIT Press, 1992.

[62] J. R. Koza and R. Poli. "Genetic Programming". In: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Ed. by E. K. Burke and G. Kendall. Boston, MA: Springer, 2005, pp. 127–164.

[63] K. Hirasawa, M. Okubo, H. Katagiri, J. Hu, and J. Murata. "Comparison between Genetic Network Programming (GNP) and Genetic Programming (GP)". In: *Proceedings of the 2001 Congress on Evolutionary Computation*. IEEE, 2001, pp. 1276–1282.

[64] S. Mabu, K. Hirasawa, J. Hu. "A Graph-Based Evolutionary Algorithm: Genetic Network Programming (GNP) and Its Extension Using Reinforcement Learning". In: *Evolutionary Computation* 15.3 (2007).

[65] X. Li, W. He, and K. Hirasawa. "Genetic Network Programming with Simplified Genetic Operators". In: *Proceedings of International Conference on Neural Information Processing*. Berlin, Heidelberg: Springer, 2013.

[66] O. Sigaud and S. W. Wilson. "Learning classifier systems: a survey". In: *Soft Computing* 11.11 (2007), pp. 1065–1078.

[67] H. H. Dam, H. A. Abbass, C. Lokan, and X. Yao. "Neural-Based Learning Classifier Systems". In: *IEEE Transactions on Knowledge and Data Engineering* 20.1 (2008), pp. 26–39.

[68] R. J. Urbanowicz and J. H. Moore. “Learning Classifier Systems: A Complete Introduction, Review, and Roadmap”. In: *Journal of Artificial Evolution and Applications* 2009.1 (2009), pp. 1–25.

[69] S. F. Smith. “A learning system based on genetic algorithms”. PhD thesis. Pittsburg: University of Pittsburg, 1980.

[70] S. W. Wilson. “A Zeroth Level Classifier System”. In: *Evolutionary Computation* 2 (1994).

[71] S. W. Wilson. “Classifier fitness based on accuracy”. In: *Evolutionary Computation* 3 (1995).

[72] P. L. Lanzi, W. Stolzmann, and S. W. Wilson, eds. *Learning classifier systems: From foundations to applications*. Berlin and Heidelberg: Springer, 2000.

[73] J. Kennedy and R. Eberhart. “Particle swarm optimization”. In: *Proceedings of ICNN’95 - International Conference on Neural Networks*. IEEE, 1995, pp. 1942–1948.

[74] R. Poli. “An Analysis of Publications on Particle Swarm Optimisation Applications”. In: *Artificial Evolution and Applications* (2007).

[75] Y. Shi and R. C. Eberhart. “Empirical study of particle swarm optimization”. In: *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99*. IEEE, 1999, pp. 1945–1950.

[76] R. P. Prado, S. Garcia-Galan, J. E. Munoz Exposito, and A. J. Yuste. “Knowledge Acquisition in Fuzzy-Rule-Based Systems With Particle-Swarm Optimization”. In: *IEEE Transactions on Fuzzy Systems* 18.6 (2010), pp. 1083–1097.

[77] R. Agrawal and R. Srikant. “Mining sequential patterns”. In: *Proceedings of the Eleventh International Conference on Data Engineering*. IEEE, 1995.

[78] P. Fournier-Viger, J. C. Lin, R. U. Kiran, Y. S. Koh, R. Thomas. “A Survey of Sequential Pattern Mining”. In: *Data Science and Pattern Recognition* 1.1 (2017).

[79] l. Schweizer, M. Zehnder, H. Wache, H. Witschel, D. Zanatta, and M. Rodriguez. “Using Consumer Behavior Data to Reduce Energy Consumption in Smart Homes: Applying Machine Learning to Save Energy without Lowering Comfort of Inhabitants”. In: *Proceedings of the 14th International Conference on Machine Learning and Applications*. IEEE, 2015, pp. 1123–1129.

[80] M. G. Gouda and A. X. Liu. “A Model of Stateful Firewalls and Its Properties”. In: *Proceedings of International Conference on Dependable Systems and Networks*. IEEE, 2005, pp. 128–137.

[81] M. Heikki, T. Hannu, and V. A. Inkeri. “Discovering Frequent Episodes in Sequences”. In: *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*. AAAI Press, 1995.

[82] M. Heikki, T. Hannu, and V. A. Inkeri. “Discovery of Frequent Episodes in Event Sequences”. In: *Data Mining and Knowledge Discovery* 1 (1997), pp. 259–289.

[83] E. Brinksma and H. Hermanns. “Process Algebra and Markov Chains”. In: *Proceedings of Lectures on Formal Methods and Performance Analysis*. Berlin, Heidelberg: Springer, 2001.

[84] P. A. Gagniuc. *Markov chains: From theory to implementation and experimentation*. Hoboken, New Jersey: John Wiley & Sons, 2017.

[85] L. R. Rabiner. “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[86] E. R. Hruschka, M. do Carmo Nicoletti, V. A. de Oliveira, and G. M. Bressan. “Markov-Blanket Based Strategy for Translating a Bayesian Classifier into a Reduced Set of Classification Rules”. In: *Proceedings of the Seventh International Conference on Hybrid Intelligent Systems*. IEEE, 2007, pp. 192–197.

[87] D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. “Dependency networks for inference, collaborative filtering, and data visualization.” In: *Machine Learning Research* 1 (2000).

[88] G. M. Bressan, V. A. Oliveira, E. R. Hruschka, and M. C. Nicoletti. “Using Bayesian networks with rule extraction to infer the risk of weed infestation in a corn-crop”. In: *Engineering Applications of Artificial Intelligence* 22.4-5 (2009), pp. 579–592.

[89] J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. “Learning Bayesian networks from data: An information-theory based approach”. In: *Artificial Intelligence* 137 (2002).

[90] J. Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[91] K. J. McGarry, J. Tait, S. Wermter and J. MacIntyre. “Rule-extraction from radial basis function networks”. In: *Proceedings of 9th International Conference on Artificial Neural Networks*. IET, 1999.

[92] D.S. Broomhead and David Lowe. “Multi-Variable Functional Interpolation and Adaptive Networks”. In: *Complex Systems* 2 (1988), pp. 321–355.

[93] C. G. Atkeson, A. W. Moore, and S. Schaal. “Locally Weighted Learning”. In: *Lazy Learning*. Ed. by David W. Aha. Vol. 11. Dordrecht: Springer, 1997.

[94] K. McGarry, S. Wermter and J. MacIntyre. “Hybrid Neural Systems: From Simple Coupling to Fully Integrated Neural Networks”. In: *Neural Computing Surveys* 2 (1999).

[95] A. Zheng and A. Casari. *Feature engineering for machine learning: Principles and techniques for data scientists*. First edition. Beijing and Boston: O’Reilly, 2018.

[96] J. McHugh. “Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory”. In: *ACM Transactions on Information and System Security* (2001).

[97] R. Sommer and V. Paxson. “Outside the Closed World: On Using Machine Learning for Network Intrusion Detection”. In: *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 2010, pp. 305–316.

[98] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard. “A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data”. In: *ACM SIGKDD Explorations Newsletter* (2004).

[99] U. Khurana, D. Turaga, H. Samulowitz, and S. Parthasrathy. “Cognito: Automated Feature Engineering for Supervised Learning”. In: *Proceedings of the IEEE 16th International Conference on Data Mining Workshops*. IEEE, 2016.

[100] J. D. Kelleher, B. A. Namee, and A. D’Arcy. *Fundamentals of machine learning for predictive data analytics: Algorithms, worked examples, and case studies*. Cambridge MA: The MIT Press, 2015.

[101] G. Helmer, J. S.K. Wong, V. Honavar, and L. Miller. “Automated discovery of concise predictive rules for intrusion detection”. In: *Systems and Software* (2002).

[102] S. S. Sivatha Sindhu, S. Geetha, and A. Kannan. “Decision tree based light weight intrusion detection using a wrapper approach”. In: *Expert Systems with Applications* 39.1 (2012), pp. 129–141.

[103] P. Nancy, S. Muthurajkumar, S. Ganapathy, S. Santhosh Kumar, M. Selvi, and K. Arputharaj. “Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks”. In: *IET Communications* 14.5 (2020), pp. 888–895.

[104] N. Stakhanova, S. Basu, W. Zhang, X. Wang, and J. Wong. “Specification Synthesis for Monitoring and Analysis of MANET Protocols”. In: *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*. IEEE, 2007.

[105] L. Li, D. Yang, and F. Shen. “A novel rule-based Intrusion Detection System using data mining”. In: *Proceedings of th 3rd International Conference on Computer Science and Information Technology*. IEEE, 2010, pp. 169–172.

[106] S. Elhag, A. Fernández, A. Altalhi, S. Alshomrani, and F. Herrera. “A multi-objective evolutionary fuzzy system to obtain a broad and accurate set of solutions in intrusion detection systems”. In: *Soft Computing* 23.4 (2019), pp. 1321–1336.

[107] N. R. Shanmugavadivu. “Network intrusion detection system using fuzzy logic”. In: *Indian Journal of Computer Science and Engineering* (2011).

[108] S. Rawat, V.P. Gulati, and A. K. Pujari. “A Fast Host-Based Intrusion Detection System Using Rough Set Theory”. In: *Transactions on Rough Sets IV*. Berlin, Heidelberg: Springer, 2005.

[109] W. Lu and I. Traore. “Detecting New Forms of Network Intrusion Using Genetic Programming”. In: *Computational Intelligence* 20.3 (2004).

[110] K. Shafi and H. A. Abbass. “An adaptive genetic-based signature learning system for intrusion detection”. In: *Expert Systems with Applications* 36.10 (2009), pp. 12036–12043.

[111] M. S. Abadeh, J. Habibi, and S. Aliari. “Using a Particle Swarm Optimization Approach for Evolutionary Fuzzy Rule Learning A Case Study of Intrusion Detection”. In: *Proceedings of Information Processing and Management of Uncertainty in Knowledge-based Systems*. 2006.

[112] S. Pan, T. Morris, and U. Adhikari. "Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems". In: *IEEE Transactions on Smart Grid* 6.6 (2015), pp. 3104–3113.

[113] W. Lee and S. J. Stolfo. "Data Mining Approaches for Intrusion Detection". In: *Proceedings of the 7th USENIX Security Symposium*. 1998.

[114] J. Luo and S. M. Bridges. "Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection". In: *Intelligent systems* 15 (2000).

[115] A. Ganesan, P. Parameshwarappa, A. Peshave, Z. Chen, and and T. Oates. "Extending Signature-based Intrusion Detection Systems With Bayesian Abductive Reasoning". In: *Proceedings of Dynamic and Novel Advances in Machine Learning and Intelligent Cyber Security (DYNAMICS) Workshop*. Vol. 10. 2018.

[116] N. Naik, R. Diao, and Q. Shen. "Dynamic Fuzzy Rule Interpolation and Its Application to Intrusion Detection". In: *IEEE Transactions on Fuzzy Systems* 26.4 (2018), pp. 1878–1892.

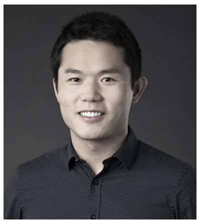
[117] S. Muggleton. "Inverse entailment and progol". In: *New Generation Computing* 13 (1995).

[118] N. Friedman. "Bayesian Network Classifiers". In: *Machine Learning* 29 (1997).

[119] A. Hofmann and B. Sick. "Evolutionary optimization of radial basis function networks for intrusion detection". In: *Proceedings of the International Joint Conference on Neural Networks*. IEEE, 2003, pp. 415–420.

[120] J. Jiang, C. Zhang, and M. Kamel. "RBF-based real-time hierarchical intrusion detection systems". In: *Proceedings of the International Joint Conference on Neural Networks*. IEEE, 2003, pp. 1512–1516.

[121] Y. Jin and B. Sendhoff. "Extracting Interpretable Fuzzy Rules from RBF Networks". In: *Neural Processing Letters* 17 (2003).



QI LIU received the B.E. degree from the School of Mechanical and Electronic Engineering, Beijing Information Science and Technology University, China, in 2016, and the M.S. degree from the Faculty of Mechanical Engineering, Karlsruhe Institute of Technology, Germany, in 2019. He is currently pursuing the Ph.D.degree at Karlsruhe Institute of Technology. His current research interests include machine learning, smart grids, smart grid security, information and communications technologies in smart grids.

PROF. VEIT HAGENMEYER received the Ph.D. degree from Universit´e Paris XI, Paris, France in 2002. He is currently a Professor of Energy Informatics in the Faculty of Computer Science, and the Director of the Institute for Automation and Applied Informatics at Karlsruhe Institute of Technology, Karlsruhe, Germany. His research interests include modeling, optimization and control of sector-integrated energy systems, machine-learning based forecasting of uncertain demand and production in energy systems mainly driven by renewables, and integrated cybersecurity of such systems.

DR. HUBERT B. KELLER is Head of the Research Area Advanced Automation Technologies (A2T) and Head of the Working Group Relaiability, Safety and Security of Software and Systems (RS4) at the Institute of Automation and Applied Informatics (IAI). Starting on 1st of July he will work as Professor of Computer Science at the DIPLOMA University of Applied Sciences. He is a principal investigator at KASTEL and responsible for security aspects at the Energy Lab 2.0 of KIT. He is leader of several research and industrial projects of BMBF, BMWi and different companies, member of several scientific groups of GI, VDI, GMA and GPP, president of Ada Germany, chair of Automotive – Safety & Security conferences from 2004 until now, and he was chair of the conferences „Reliable Software Systems“ 2000 and 2013. His research interests are Real Time Systems, Software Engineering, Machine Intelligence, intelligent Sensor Systems and Networks and safe and secure Software. He is lecturer for Computer Engineering at KIT, author of the books Maschinelle Intelligenz of 2000 and Technical Safety – An Attribute of Quality of 2018, co-author of the NE153, Automation Security 2020 –Design, Implementation and Operation of Industrial Automation Systems, and got several best paper awards.

• • •