Modified Intrusion Detection Tree with Hybrid Deep Learning Framework based Cyber Security Intrusion Detection Model

Majed Alowaidi
Department of Information Technology, College of Computer and Information Sciences
Majmaah University, Majmaah, 11952, Saudi Arabia

***Abstract*—In modern era, the most pressing issue facing modern society is protection against cyberattacks on networks. The frequency of cyber-attacks in the present world makes the problem of providing feasible security to the computer system from potential risks important and crucial. Network security cannot be effectively monitored and protected without the use of intrusion detection systems (IDSs). DLTs (Deep learning methods) and MLTs (machine learning techniques) are being employed in information security domains for effectively building IDSs. These IDSs are capable of automatically and timely identifying harmful attacks. IntruDTree (Intrusion Detection Tree), a security model based on MLTs that detects attacks effectively, is shown in the existing research effort. This model, however, suffers from an overfitting problem, which occurs when the learning method perfectly matches the training data but fails to generalize to new data. To address the issue, this study introduces the MIntruDTree-HDL (Modified IntruDTree with Hybrid Deep Learning) framework, which improves the performance and prediction of the IDSs. The MIntruDTree-HDL framework predicts and classifies harmful cyber assaults in the network using an M-IntruDtree (Modified IDS Tree) with CRNNs (convolution recurrent neural networks). To rank the key characteristics, first create a modified tree-based generalized IDSs M-IntruDTree. CNNs (convolution neural networks) then use convolution to collect local information, while the RNNs (recurrent neural networks) capture temporal features to increase IDS performance and prediction. This model is not only accurate in predicting unknown test scenarios, but it also results in reduced computational costs due to its dimensionality reductions. The efficacy of the suggested MIntruDTree-HDL schemes was benchmarked on cybersecurity datasets in terms of precisions, recalls, fscores, accuracies, and ROC. The simulation results show that the proposed MIntruDTree-HDL outperforms current IDS approaches, with a high rate of malicious attack detection accuracy.**

Keywords*—Cybersecurity; IntruDTree model; convolution recurrent neural network (CRNN); MIntruDTree-HDL; deep learning

I. Introduction

The Internet is increasing being intertwined with social lives and revolutionising the way people learn or work, but is also getting exposed to serious security lapses [1] and identifying serious and new threats is a critical issue that must be addressed immediately. Cybersecurity is term used to represent technologies and processes designed to protect computers, networks, programmes from unknown attacks or unauthorised accesses or alterations or destructions of data. The term information security is also used interchangeably with cybersecurity where the former recognizes human functions in security processes while the latter adds extra dimensions in focusing on possible targets [2]. However, since, it focuses on ethical components of society it has important ramifications. Techniques need to protect data in all forms including its processing, transmissions physical/virtual storages in information technologies by setting up higher security levels achieved in adopting professional measures associated with security [3]. Cybersecurity is thus prevention measures from unwanted accesses, usage, disclosures, or modifications of data using computer systems or networks.

Network security systems [4] are made up of two parts namely security systems and computers where Firewalls, antivirus software, and IDSs are all included in the overall picture. External and internal invasions are examples of security breaches. IDSs aid in the detection, determination, and identification of any illegal system activities executed by attackers [5]. Because of their ability to identify zero-day threats, they are enticing. Another benefit is that typical activity profiles can be customized based on the characteristics of systems or applications or networks which makes it difficult for attackers to execute operations unnoticed [6]. Additionally, the data that anomaly-based approaches identify (new assaults) may be leveraged to develop abuse detection signatures. Because previously undiscovered system actions might be classified as anomalies, the fundamental downside of these techniques is their generation of higher false alarms [7]. AIDSs (Anomaly based IDSs), on the other hand, evaluates network activities for trends, automatically construct data-driven models that profile usual behaviours and detect deviations in case of irregularities. The primary advantage of AIDSs over signature based IDSs are their ability to trace previously undisclosed vulnerabilities or cyber-threats [8]. However, treating previously undetected system actions as anomalies may also produce higher false alarms where MLTs can be employed to handle these issues.

Traditional MLTs fall into shallow learning groups as they do not adequately address attack classifications in real world network applications as they concentrate lesser on feature engineering or feature selections. Multi-classification attack detection tasks become less accurate as dataset sizes grow [9]. As a result, DLTs have lately been proposed as a way to improve the intelligence of IDSs, despite the lack of research

to compare such MLTs to publically available datasets [10]. These issues were the base motivation for this work which proposes a hybrid IDSs based on CNNs with evaluations of its efficacy [11].

The remainder of the research is structured as follows: The second section examines some of the most modern strategies for identifying cyber threats. The proposed technique is presented in Section III. Section IV summarises the findings and discusses them. The limitations are highlighted in Section V. The conclusion and future efforts are discussed in Section VI.

II. Related Work

IDSs generally identify malicious activities on networks while monitoring them for analyzing or discovering security risks. Several cybersecurity studies have been conducted with the aim of identifying or preventing cyber-attacks or security breaches. This section details about studies related to strategies for avoiding cyber-attacks.

Martnez Torres et al. [12] pioneered the use of MLTs in cybersecurity, defining several kinds of models based on (1) structures (network based/non-network based), (2) learning methods (supervised/unsupervised), and (3) complexities. Their descriptions were useful for further researchers on usage of MLTs in cybersecurity. Yin et al [13] proposed DLTs for IDSs where RNNs were used. The scheme called RNN-IDS showed that it was suited for building highly accurate classifications were in experimental results, and that their performances in binary/multi-class classifications surpassed performances of traditional MLTs. Kim et al. [14] also suggested DLT based IDSs which was tested on KDD Cup 1999 datasets. The scheme used LSTMs (Long Short Term Memories) with RNNs to learn. The study's results confirmed the success of DLT based IDSs in detecting malicious activities on networks. Al-Qatf et al [15] suggested IDSs based on STL (self-taught learning) frameworks and successful DLTs where the study's suggested method learnt features and reduced dimensionalities for minimizing training/testing execution times while enhancing prediction accuracies of assaults by SVMs (support vector machines). The study's suggested STL-IDSs technique improvised network IDSs while also introducing novel IDSs.

Khan et al. [16] proposed a pattern recognition technique for anticipating Denial of service (DoS) assaults with a higher prediction level. The method through DoS attacks is detected. DoS attacks are extremely serious assault that puts an organization's IT resources at risk by flooding them with fake messages or numerous requests from unauthorised users. Lekha et al [17] offered a broad overview of DMTs (Data Mining techniques) and cyber crimes in banking applications. According to the study, K-Means clustering, Influenced Association Classifiers, and J48 Prediction Trees combinations enabled complete, integrated, and precise cyber crime predictions in the banking sector. Law enforcements need to be strong to combat and prevent terrorism. Mitchell et al [18] developed probabilistic models based on stochastic Petri nets to identify behaviour of malicious nodes in CPSs (cyber physical systems) and IDRSs (intrusion detection and response systems) and thus respond to these real time malicious events. Three different mechanisms for time-based IDSs were presented by Zimmer et al [19] where execution of illegitimate instructions in real-time CPSs were specifically identified using static timing analyses. Li et al [20] used CNNs with gated recurrent units for their suggested novel IDSs based on DLTs for industrial CPSs. The study used integrated learning in their architecture by allowing multiple industrial CPSs to work together and thus develop comprehensive new models of IDSs. Dutta et al. [21] established robust anomaly detections using semi-supervised MLTs that traced real time assaults. The proposed scheme applied DNNs (deep neural networks) using reconstruction errors for its detections. Their tests on the SWaT dataset show its efficacy by achieving AUC value of 0.9275 and better than other known anomaly detection algorithms. Sarker et al [22] introduced IntruDTree (Intrusion Detection Tree) security model based on MLTs that first examined the importance of security factors before constructing trees from fundamental features for generalizing IDSs. To evaluate performances of their resultant security models, the study compared findings of IntruDTrees with many classical MLTs including NBs (Naive Bayes), LRs (Logistic regressions), SVMs and KNNs (k-nearest neighbours).

IntruDTree did a reasonable job of detecting attempts at intrusion; but, it had issues with overfitting while it was learning, and it was unable to generalize what it had learnt so that it could apply it to new data. As a consequence of this, the goal of this effort is to improve classifier performances while at the same time recognizing cyber intrusions in real time. It increases the learning rates of IDSs, which in turn leads to an effective improvement in the performances and predictions of the system.

III. Proposed Methodology

MIntruDTree-HDL framework is suggested in this research work for predicting and classifying harmful cyberattacks in networks. The ranking of important features are done by constructing trees in IntruDTree framework. The performances of predictions are enhanced by CNN's convolutions which gather local information, while RNNs acquire temporal aspects. On the completion of the tree from training data, tests are used to validate the suggested framework. The proposed framework reduces computational complexities by reducing feature dimensionalities resulting in minimizing over fits of data and thus lays the base for improved prediction accuracies of unknown test cases. The contributions of this study can be summarized as:

- IDSs based on MLTs are presented emphasizing the importance of high dimensional security features.
- Proposing IntruDTree framework for ranking of security features based on their importance and subsequently uses them to build generalized trees encompassing chosen features.
- Increasing predictive performances of IDSs by using CNNs for collect local information and RNNs for capturing temporal features.
- Testing the IntruDTree framework for evaluating its performances.

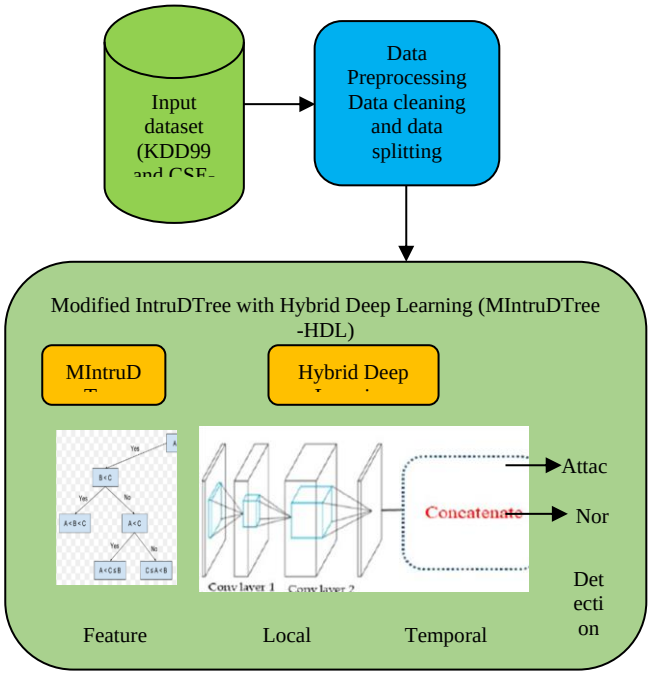

Fig. 1. The Suggested Methodology of this Research Work.

The experiment results of the suggested IntruDTree framework shows that it outperforms prior schemes in its detection previously unseen test cases of cyber intrusions. Fig. 1 depicts the suggested methodology.

A. Materials and Methods

The suggested IntruDTree framework is presented where security datasets are processed using a multitude of steps. Raw data is pre-processed, feature's relevance are assessed and ranked for building generalized trees. The stages of the proposed methodology are detailed below in this section.

B. Exploring Security Dataset

Security datasets contain collections of data records which include many securities related information that can be used for constructing data-driven IDSs for cyber security. Understanding raw cybersecurity data and security event patterns are critical for detecting malicious irregularities or behaviors [22]. This work used intrusion detection dataset from Kaggle (largest machine learning and data science community) with two types of classes namely normal and anomalous. The dataset had 41 features with 3 qualitative features (protocol type, service, f delay). The other features were quantitative. The dataset's security aspects are listed in Table I. The dataset had 25,000 examples obtained from simulated military network intrusion settings where US Air Force local area networks were modelled including TCP/IP dumps of data from networks. Networks were in parallel to cyber environments and subjected to a wide range of cyber-attacks or anomalies.

Encoding of features: As previously detailed, dataset had both numeric and notional values of security issues. Though most features were numerically valued, nominally values also existed (protocol type, service, f delay along with classes [anomaly, normal]). All nominal values were translated into vectors for fitting them into MLTs where "Label Encoding" was used though most studies had used "One Hot Encoding". The values of tcp, udp, icmp, udp, icmp were label encoded for converting them into vectors.

TABLE I. FEATURES AND THEIR DATA TYPES

Feature name	Value Type	Feature Name	Value Type
dst_host_srv_count	Integer	Same_srv_rate	Float
flag	Nominal	dst_host_same_srv_rate	Float
srv_serror_rate	Float	dst_host_srv_serror_rate	Float
protocol_type	Float	count	Integer
dst_host_same_src_port_rate	Nominal	logged_in	Integer
error_rate	Float	dst_host_srv_diff_host_rate	Float
dst_host_srv_rerro_rate	Float	src_bytes	Integer
srv_rerror_rate	Float	service	Nominal
dst_host_count	Float	dst_host_rerror_rate	Float
srv_count	Integer	dst_host_diff_srv_rate	Float
serror_rate	Integer	wrpmg_fragment	Integer
srvdiff_host_rate	Float	num_compromised	Integer
hot	Integer	dst_bytes	Integer
duration	Integer	diff_srv_rate	Float
root_shell	Integer	is_guest_login	Integer
urgent	Integer	land	Integer
su_attempted	Integer	num_failed_logins	Integer
num_file_creation	Integer	num_root	Integer
num_access_files	Integer	num_shells	Integer
is_host_login	Integer	num_outbound_cmds	Integer

Table I displays unique data distributions of outlined security methods. This work prepared raw datasets from above-mentioned features for its proposed IDSs based on DLTs. Processes and ranks of security features were based on and targeted DLT requirements and data patterns were constructed for ensuring anomalies and intrusions could be traced by intelligent cyber security services.

C. Preparation of Data from Raw Security Data: Data Preparations Include Encoding Feature and Scaling them According to Parameters of Intrusion Datasets

Feature scaling: Data normalizations are also called feature scaling in data pre-processing. Security feature may have a range of values and need to be scaled or normalized to acceptable ranges. This study used Standard Scaler to equalize security characteristics with mean values of 0 and standard deviations of 1. Subsequently, these normalized values were further analyzed while building the security model.

D. Determining Feature Importance and Ranking

On completion of investigations and preparations, relevance scores of security features were obtained and ranked based on their importance and to choose critical features for future processing. Their values varied between 0 and 1 where

0 indicates that the model's output has no relation to the feature, whereas a value of 1 shows that the model's output has a direct relationship with the feature. Thus, the "purity" of attributes were determined. Gini Indices are well-known measures for assessing node's impurity in statistics and data mining that generally judge frequency of random elements. It is the probability of mistakenly categorizing a randomly selected element in a security dataset based on class distribution of the dataset. In binary splits, Gini Indices of nodes n can be expressed as

$$I_{G(n)} = 1 - \sum_{i=1}^{c} p_i^2 \quad (1)$$

$$\Delta I_{G(n_p)} = I_{G(n_p)} - p_l I_{G(n_l)} - p_r I_{G(n_r)} \quad (2)$$

where pi is the likelihood of elements being categorized as belonging to certain security classes and pl and pr are percentages of examples in nodes np allocated to child nodes nl and nr, respectively. Hence, Gini impurity equations determine feature's decreases of impurity. Greater ability of attributes to eliminate impurities improves its significance. Security features are sorted based on their computed importance and after evaluating the features, this work selects top n security features based on their relevance score values in order to develop effective tree-based security models by employing the n features selected.

E. Design of M-IntruDTree (Modified IntruDTree)

The chosen security characteristics are used to build trees for taking decisions by the intelligent data-driven IDS. In this model, all features of the dataset are not chosen instead only security features are chosen based on their relevance scores and ranks. A root is formed first, followed by the construction of connected branches of the tree in which the training dataset is divided into smaller sub-groups. This model properly matches the training data, however fails on generalizations of test data. Models memorizing training data noises tend to miss important patterns as over fits occur. DTs (Decision Trees) perform well on training data, but fail on unknown test data and hence to effectively over the issue of over fits, this study uses I-RLRs (Inductive Rule Learning Rates). Gini Indices are used to identify root node attribute in each level and gradually the tree is built with lower Gini values resulting in adding required counts of branches encompassing internal/leaf nodes with corresponding arcs or connecting edges. Labels on internal nodes are based on defined or selected security criteria while node's leaves are labelled with security features which can be one of the two: anomaly or normal. Fig. 2 displays multi-level trees with terminals or node's leaves with defined labels. This work's IntruDTree concentrates on achieving two main objectives which are reducing dimensionality of features based on their evaluated ranks or relevance and generating multiple-level trees from chosen critical features. Fig. 3 displays IntruDtree's considered IDS features like f latency, services, durations, and logged in.

- I-RERs (Inductive Re-substitution Error Rate): The fundamental idea underlying I-RERs is that instead of writing complete concept descriptions first and then trimming them, individual phrases are cut as soon as they are written which ensures the algorithm can eliminate training instances like trimming even before learning subsequent phrases and preventing these examples from influencing success of learning clauses. Algorithm 1 depicts pseudo-code of this approach. Traditionally current collection of training examples is separated into growths (generally 2/3) and pruned (typically 1/3). However, due to the growth of collections, only one line is learned. In greedy approaches, literals from phrases are removed till deletions which can diminish validity of sentences are maintained on pruning sets. These sets are then used to derive clauses where prediction accuracy of trimmed clauses fall below empty clauses (i.e., clause with body fail), clauses are removed from concept descriptions, and I-RERs return learnt clauses. As a result, accuracy of trimmed phrases on pruning sets also serve as stopping criteria. The asymptotic complexity of I-RERs is O (n log2 n), where n is the size of the training set. The cost of adding one clause in RERs is O (n log n), since the cost of picking sentences is Θ(log n) literals. As a consequence, O(n) instances are tested against fixed sets of criteria. I-RERs consider cutting every literal in phrases. Resulting in evaluating (log n) literals on (n) samples in pruning sets at most O (log n) times until the last phrase is discovered. hence, eliminating one clause costs O (n log2 n). If I-RERs finish with identified constant sizes, the overall cost is also O. (n log2 n) which is far less expensive than creating over fits determined by (n2 log n) under the same assumptions. For efficiency of computing costs, nodes must contain few values.
- R(t), re-substitution error rates at nodes (t) and done only once.
- R(Tt), re-substitution error rates at branch emerging from nodes (t) can be modified since Tt varies on pruning.
- |Tt|, leaf node counts on branches from nodes (t) and may change on pruning.

To calculate re-substitution error rates R(t), data points in each class that arrive at node t are divided by the data point counts in each class that arrive at node t. Assuming the fraction of points in classes are utilized to build class priors, then R(t) can be computed. After pruning, both re-substitution error rates of branches coming out of node t and the leaf nodes counts on the branch coming out of node t fluctuate. These variables will need to be adjusted because the leaf nodes counts would have dropped after pruning. To be more specific, all values for branch's ancestor nodes must be changed where R(Tt) and |Tt| may be computed using a recursive process.

Counts of leaf nodes in branches of nodes (t) are determined using bottom up sweeps of constructed trees. These leaf nodes counts are equal to the sum of counts of leaf nodes on the right child nodes and counts of leaf nodes on the left child nodes. The leaf node counts for child nodes are determined before determining the counts of parent nodes in bottom up sweep operations. The values of R(Tt) are also equal to the sum of two child node values of (t). These three values are the base for determining ratio g(t) and identifying

weakest connections. The new is the comparable ratio at the weakest connection α which guarantees that sequences of α are obtained in pruning are strictly increasing. When there are many weakest links, for example, if g_k (t¯_k)=g_k ((t')¯_k), then define:

$$T_{(k+1)}=T_k-T_{(t\bar{}_k)}-T_{(((t')\bar{}_k),)} \quad (3)$$

Branch nodes can be nested or share modes where pruning procedure resulting in sequence of nested sub-trees can be:

$$T1>T2>T3>\cdots>t1 \quad (4)$$

Algorithm 1 describes the general procedure for constructing an IntruDTree. Given a training intrusion dataset, DS = fX1, X2,..., Xmg, where m denotes the amount of the data. n-dimensional characteristics are used to represent each instance. The training data is also divided into various cyber-attack classes. CA stands for fnormal and anomalyg. An IntruDTree, a rule-based classification tree connected with DS, is the result. Fig. 2 depicts single feature's rules, when flag's value is RSTR, it implies anomalous. Multi-aspect rules for flag values can be SF implies ftb service and duration value of four implies anomalous. By traversing the resulting IntruDTree, multiple rules for security can be retrieved based on which the final outcome would result in normal or abnormal.

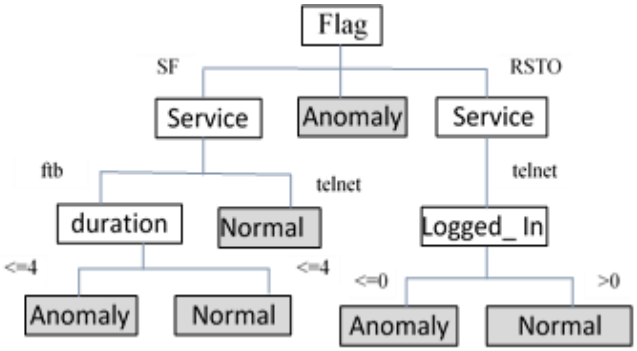

Fig. 2. M-IntruDTree Built from Features.

F. Hybrid Deep Learning Model

With a data processing architecture, hybrid deep learning focuses on tackling actual ID challenges. By combining a CNN and an RNN model, hybrid deep learning reduces these issues [23]. The main structure of the HDL that occurs here is the source of the experiment. Fig. 1 shows the suggested model in further detail. A CNN contains two main components, according to the HDL: I a feature extractor; and (ii) a classifier. Convolution and pooling layers are the two layers that make up the feature extractor. The feature map, which is the extracted result, is used as the input to the classification's second component. In this method, CNN gains a thorough understanding of the local characteristics. However, it has a flaw in that it ignores the temporal relationship between significant traits. After the CNN layers, recurrent layers were added to capture both spatial and temporal data more effectively. This method effectively handled the disappearing and inflating gradient difficulties, enhancing the capacity to record and learn from variable extent sequences and spatial and temporal correlations. In the HDL model, CNNs are hybridized as RNNs, and inputs are initially processed by CNNs, after which the outputs are relayed via recurrent layers to form sequences at time steps, allowing capture of both spatial and temporal characteristics. As with AIDS, the bulk of traffic is categorized based on its behavior, which should not be biased or conflicting with the IP address, hence the IP address characteristics were also eliminated. Training sets were utilized for training while validation sets were used for fast prototype evaluations while training and testing sets evaluated the final model. Moreover, it was noticed that the dataset had too many instances of typical network's traffic, which can impact classifications the model.

Algorithm 1. Pseudocode of MIntruDTree

Data: Dataset: D = X_1, X_2, ..., X_m // Occurrences of X_i have features and CIs (cyber Intrusions) class information

Result: MIntruDTree

1 Procedure MIntruDTree (D, features_list, CIs);
2 //generate feature significance scores
3 $imp_{score} \leftarrow$ *compute score features_list*
4 //Choose significant features
5 imp_ features_list ← ChosenFeatures(features_list, imp_{score}, n)
6 TreeGens(D, imp_ features_list, CIs)
7 N ← createNodes() //create tree's root node
8 if all instances in D belong to the same class of CIs **then**
9 return N as leaf node labelled with class CI.
10 end
11 if imp_features_list is null **then**
12 return N as leaf node labeled with majority class of D; // majority votes
13 end
14 identify features with highest precedence feature F_{split} for dividing and assigning F_{split} to node N.
15 for each feature's value val ∈ F_{split} **do**
16 create subset D_{sub} of D with val.
17 if $D_{sub} \neq \emptyset$ **then**
18 attach node returned by TreeGens(D_{sub}, {imp_ f eatures_list - F_{split}}, CIs)) to node N;
19 end
20 // Inductive re-substitution error rates
21 calculate R(Tt) and |Tt|
22 modify nested subtrees T1>T2>T3>···>t1
23 attach leaves labeled with majority class of D to node N;
24 end
25 return N

IV. Results and Discussion

The experimental outcomes of this work are briefly discussed and reported in this section. First, we'll set up our tests to assess the suggested MIntruDTree-HDL cyber security model, and then we'll talk about the outcomes. This project uses the KDD99 and CSE-CIC-DS2018 datasets and is written in Java. These datasets are chosen based on a range of characteristics, including the amount of samples, attributes, and classifications. To compute various performance measures, TPs (true positives), FPs (false positives), TNs (true negatives), and FNs (false negatives) are measured. Precision, defined as the proportion of relevant retrieved instances, was the original performance metric. The second performance

parameter was recall, which was defined as the percentage of relevant instances. Despite their usually contradicting character, the assessments of accuracy and recall are both critical in evaluating the efficiency of a prediction approach. As a result, these two measures may be combined with equal weights to form the F-measure, which is a single metric. The last performance criterion was the accuracy measure, which was defined as the fraction of correctly predicted occurrences compared to all anticipated instances.

Precisions are defined as proportions of accurately identified positive observations against all predicted positive observations.

$$\text{Precision} = TP/(TP+FP) \quad (5)$$

The ratio of accurately detected positive observations to total observations are termed recalls.

$$\text{Recall} = TP/(TP+FN) \quad (6)$$

F-measures can be defined as weighted averages of Precisions and Recalls and hence they consider FPs and FNs.

$$\text{F1 Score} = 2*(\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision}) \quad (7)$$

Accuracies are computed in terms of positives and negatives as shown below:

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN) \quad (8)$$

This work selected features that satisfies threshold values of t = 0.02 which resulted in the selection of 14 features based on importance scores of Table II.

Fig. 3 illustrates precision comparison values between the proposed and existing method for detecting the cyber-attacks. Therefore, the results verify that the ranking the features using M-IntruDTree can be effective in extracting the given data. Thus the proposed model has the number of useful features does not affect the performance of the jointly learnt features transformation very much. From the given two dataset, KDD99 has high detection rate than the other CSE-CIC-DS dataset.

TABLE II. KDD99 DATASET'S TOP RANKED FEATURES WITH THEIR IMPORTANCE SCORES

Ranking	Security Feature Name	Importance Score
01	src_bytes	0.258093
02	dst_bytes	0.129825
03	flag	0.073396
04	dst_host_same_srv_rate	0.059504
05	dst_host_srv_count	0.053630
06	dst_host_diff_srv_rate	0.046281
07	diff_srv_rate	0.041144
08	count	0.040548
09	same_srv_rate	0.036620
10	protocol_type	0.31650
11	dst_host_same_src_port_rate	0.025566
12	service	0.023904
13	serror_rate	0.023188
14	logged_in	0.020901

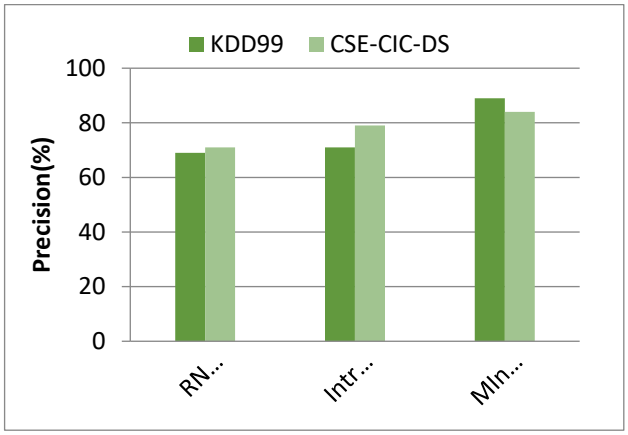

Fig. 3. Precision Comparison Results between the Proposed and Existing Method for Detecting the Cyber.

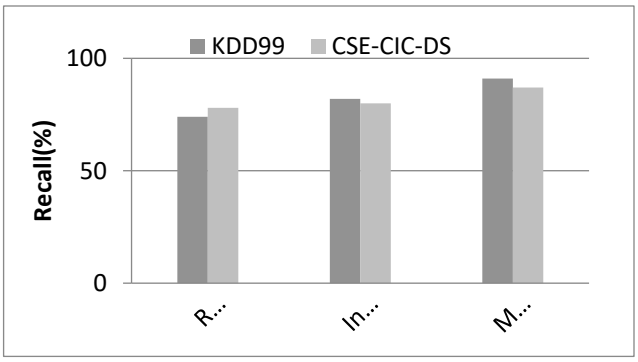

Fig. 4. Comparative Recall Values of the Proposed and Existing Methods for Detecting Cyber Attacks.

Fig. 4 illustrates recall comparisons between the proposed and existing method for detecting the cyber-attacks. Thus the results show that the proposed method gives the high recall results of 91% whereas the existing technique has less recall results such as IntruDTree method metric has 82%, and the RNN-IDSs method metric has 74% for KDD99 data. On the other hand, proposed method gives the high recall results of 87% whereas the existing technique has less recall results such as IntruDTree method metric has 80%, and the RNN-IDSs method metric has 78% for CSE-CIC-DS. Fig. 5 depicts F-measure comparative values of the proposed and existing methods for identifying cyber assaults.

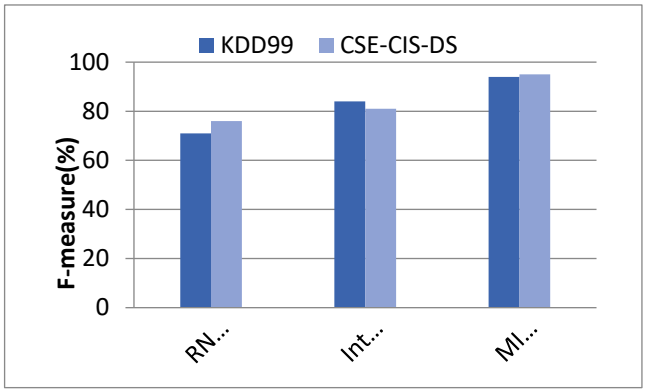

Fig. 5. F-measure Comparative Values of the Proposed and Existing Methods for Identifying Cyber Assaults.

The findings show that the suggested MIntruDTree outperforms existing attack detection strategies in terms of F-measure values.

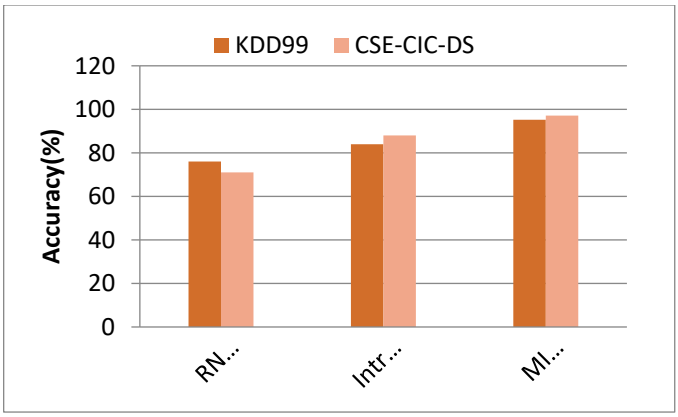

Fig. 6. Comparative Accuracies of Proposed and Existing Methods in Detecting Cyber Attacks.

Fig. 6 depicts the accuracy comparison findings between the proposed and current methods for detecting cyber assaults. In addition, the suggested method's average accuracy of classification rates (in per cent) is shown in comparison to other approaches employing IntruDTree and RNN-IDSs across ten separate runs. As the results show, the suggested technique has greater detection accuracies than previous methods in most circumstances. Furthermore, the rank of feature extraction algorithms is presented based on the detection accuracy attained for a specific dataset. The findings show that the suggested MIntruDTree-HDL approach ranked top when compared to other methods.

V. Limitations

IDSs also have certain limitations hence this work integrates two techniques for overcoming shortcomings of IDSs where the suggested method benefits by taking advantage of the used approaches. The MIntruDTree-HDL framework is presented to improve learning rates of IDSs and thus effectively enhance its performances and predictions.

VI. Conclusion

A modified IDSs tree (MIntruDTree) and a Hybrid Deep Learning (HDL) security model are discussed in this work where important security factors were first prioritized, and subsequently tree-based generalized IDSs were created based on the essential characteristics that were chosen. This was done to ensure that the security model was effective in terms of prediction accuracy for unknown test conditions, as well as efficient by reducing the computational cost of generating the future MIntruDTree-like model by processing fewer features. Following the CNN layers, we added recurrent layers to better capture both spatial and temporal data. We hoped to overcome the vanishing and growing gradient problems with our strategy, improving the ability to collect spatial and temporal correlations and learn effectively from them. The primary motivation for developing IDSs based on DLT categorization. The suggested IDSs aid in the reduction of computing complexity and improve the accuracy and DRs of IDSs. Known classification metrics were used to evaluate both traditional MLTs and DLTs (DRs, Accuracies, Precisions, Recalls, and F1-scores). The results of the simulations reveal that the proposed MIntruDTree with HDL may successfully calcify harmful attack events. In the KDD99 dataset, the total accuracy of normal and other forms of assaults is approximately 95.24 per cent, while in the CSE-CIC-IDS2018 data, it's around 97.12 per cent. On the basis of the results of the simulation, we can conclude that it is possible to develop an effective security solution against harmful attacks by utilizing a MIntruDTree-based hybrid distributed ledger technology (DLT). This methodology is improved further to include additional deep learning methods and a feature extraction strategy if there are different identification problems in actual datasets currently being used. This is done in preparation for the possibility that these problems may exist.

Acknowledgment

Dr Majed Alowaidi would like to thank the Deanship of Scientific Research at Majmaah University for supporting this work under Project No. R-2022-284.

References

[1] Kott, A. (2014). Towards fundamental science of cyber security. In Network science and cybersecurity (pp. 1-13). Springer, New York, NY.

[2] Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A survey of deep learning methods for cyber security. Information, 10(4), 122.

[3] Lee, W., Stolfo, S. J., & Mok, K. W. (2000). Adaptive intrusion detection: A data mining approach. Artificial Intelligence Review, 14(6), 533-567.

[4] Abou Messaad, M., Jerad, C., & Sikora, A. AI Approaches for IoT Security Analysis. Intelligent Systems, Technologies and Applications: Proceedings of Sixth ISTA 2020, India, 1353, 47.

[5] Buczak, A. L., & Guven, E. (2015). A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications surveys & tutorials, 18(2), 1153-1176.

[6] Chander, B., & Kumaravelan, G. (2021). Cyber security with AI—Part I. In The" Essence" of network security: An end-to-end panorama (pp. 147-171). Springer, Singapore.

[7] Thakur, K., Qiu, M., Gai, K., & Ali, M. L. (2015, November). An investigation on cyber security threats and security models. In 2015 IEEE 2nd international conference on cyber security and cloud computing (pp. 307-311). IEEE.

[8] Kantarcioglu, M., & Xi, B. (2016, October). Adversarial data mining: Big data meets cyber security. In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security (pp. 1866-1867).

[9] Singhal, A. (2007). Data warehousing and data mining techniques for cyber security (Vol. 31). Springer Science & Business Media.

[10] Mitchell, R., & Chen, I. R. (2014). A survey of intrusion detection techniques for cyber-physical systems. ACM Computing Surveys (CSUR), 46(4), 1-29.

[11] Handa, A., Sharma, A., & Shukla, S. K. (2019). Machine learning in cybersecurity: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9(4), e1306.

[12] Martínez Torres, J., Iglesias Comesaña, C., & García-Nieto, P. J. (2019). Machine learning techniques applied to cybersecurity. International Journal of Machine Learning and Cybernetics, 10(10), 2823-2836.

[13] Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. Ieee Access, 5, 21954-21961.

[14] Kim, J., Kim, J., Thu, H. L. T., & Kim, H. (2016, February). Long short term memory recurrent neural network classifier for intrusion detection.

In 2016 international conference on platform technology and service (PlatCon) (pp. 1-5). IEEE.

[15] Al-Qatf, M., Lasheng, Y., Al-Habib, M., & Al-Sabahi, K. (2018). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. Ieee Access, 6, 52843-52856.

[16] Khan, M. A., Pradhan, S. K., & Fatima, H. (2017, March). Applying data mining techniques in cyber crimes. In 2017 2nd International Conference on Anti-Cyber Crimes (ICACC) (pp. 213-216). IEEE.

[17] Lekha, K. C., & Prakasam, S. (2017, August). Data mining techniques in detecting and predicting cyber crimes in banking sector. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) (pp. 1639-1643). IEEE.

[18] Mitchell, R., & Chen, R. (2013). Effect of intrusion detection and response on reliability of cyber physical systems. IEEE Transactions on Reliability, 62(1), 199-210.

[19] Zimmer, C., Bhat, B., Mueller, F., & Mohan, S. (2010, April). Time-based intrusion detection in cyber-physical systems. In Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems (pp. 109-118).

[20] Li, B., Wu, Y., Song, J., Lu, R., Li, T., & Zhao, L. (2020). DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. IEEE Transactions on Industrial Informatics, 17(8), 5615-5624.

[21] Dutta, A. K., Negi, R., & Shukla, S. K. (2021, July). Robust multivariate anomaly-based intrusion detection system for cyber-physical systems. In International Symposium on Cyber Security Cryptography and Machine Learning (pp. 86-93). Springer, Cham.

[22] Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). Intrudtree: a machine learning based cyber security intrusion detection model. Symmetry, 12(5), 754.

[23] Abroyan, N. (2017, August). Convolution and recurrent neural networks for real-time data classification. In 2017 Seventh International Conference on Innovative Computing Technology (INTECH) (pp. 42-45). IEEE.