Hybrid Ensemble Network Intrusion Detection

DR. SUBBA RAO KOLAVENNU1, G RAKESH REDDY2, K. VIGNESHWAR REDDY3, P. SRAVYA SREE[4], CH. SRIRAM REDDY[5]

[1] *Computer Science & Engineering (CS) (JNTUH) Sphoorthy Engineering College. (JNTUH)*

[2] *Computer Science & Engineering (CS), (Assistant Professor), Sphoorthy Engineering College (JNTUH)*

[3, 4, 5] *Computer Science & Engineering (CS) (B. Tech, JNTUH) Sphoorthy Engineering College (JNTUH)*

Abstract*— In recent years, the exponential growth of networked devices has revolutionized everyday life. However, it has also attracted the attention of cybercriminals, which has caused an increase in the number and sophistication of attacks against these devices. Network Intrusion Detection System (NIDS) has become an essential part of network applications to detect such attacks. However, network devices generate huge amounts of high-dimensional data, which makes it difficult to detect known and unknown attacks with greater accuracy. Additionally, the complicated nature of network data complicates the NIDS feature selection procedure. In this work, Our proposal is an autonomous feature selection and two-stage hybrid ensemble learning machine learning-based intrusion detection system. The suggested system performs automatic feature selection based on the capacity of four distinct machine learning classifiers to identify the most important features. The hybrid ensemble learning algorithm is designed in two stages: the first stage is built using classifiers that are created by modifying the One-vs-One framework, and the second stage is built using classifiers that are created by combining different attack classes. In comparison to other similar studies found in the literature, the evaluation of the proposed framework on two well-referenced datasets for both wired and wireless applications demonstrates that the two-stage ensemble learning system paired with an automatic feature selection module has superior attack detection ability.

I. INTRODUCTION

An Intrusion Detection System (IDS) is a network security technology originally created to detect the exploitation of a vulnerability against a target application or computer. IDS is also a listen-only device. The IDS monitors the traffic and reports the results to the administrator. With this growing popularity, the security and privacy aspects of both wired and wireless networks have gained considerable importance in recent years. These security issues are usually addressed by a Network Intrusion Detection System (NIDS), which sits at the center of the network and continuously monitors incoming and outgoing packets to detect attacks. There are several challenges associated with designing an effective NIDS based on machine learning. Network devices produce large volumes of complex high-dimensional data, which makes sense of the very definition of big data. The high dimensionality itself creates several new challenges. For high-dimensional data, selecting too many elements can lead to problems such as the curse of dimensionality. Accurate automatic feature selection therefore becomes very important for large-scale large-scale network data, as manual feature selection can be a daunting task that may lead to incorrect feature selection or misspecification of rules. Automatic feature selection for network data remains an active research topic.

II. LITERATURE SURVEY

We have studied the existing ventures and at last thought of making essential adjustments for getting the most recent edition.

III. EXSISTING SYSTEM

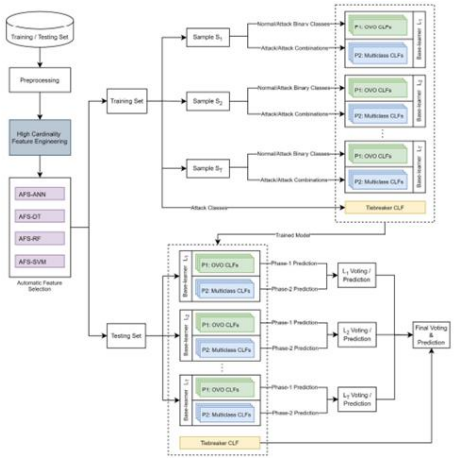
In the literature, they presented an intrusion discovery system using an ANN classifier. It's an intelligent system that first performs point bracket grounded on information gain and correlation. point reduction is also performed by combining the values attained from both information gain and correlation using a new approach to identify useful and useless features. These reduced features are also fed to a feedforward neural network for training and testing on the KDD99 dataset. Preprocessing of the KDD- 99 dataset was performed to homogenize the number of cases of each class before training. Their system also behaves intelligently to classify the test data into attack andnon-attack classes. Their point-limited system

aims to achieve the same degree of performance as a normal system. Their system is tested on five different test data sets, and individual and average results of all data sets are reported. A comparison of the proposed system with and without point reduction is made in terms of colorful performance criteria.

IV. PROPOSED SYSTEM

We propose a machine learning-based NIDS with two-stage hybrid Ensemble learning and automatic feature selection. The proposed framework uses four different machine learning classifiers to perform automatic feature selection based on their ability to detect the most significant features. The two-phase hybrid ensemble learning algorithm consists of two learning phases, with the first phase constructed using classifiers constructed from adapting the One-vs-One framework, and the second phase constructed using classifiers constructed from combinations of attack classes. The proposed framework has been evaluated on two well-referenced datasets such as NSL-KDD and AWID for both wired and wireless applications. The NSL-KDD dataset was used to evaluate the wired application and the AWID dataset was used to evaluate the wireless application.

V. SYSTEM ARCHITECTURE



VI. FEASIBILITY STUDY

A feasibility study evaluates the practicality of a project or system. As part of a feasibility study, an objective and rational analysis of a potential venture or business is conducted to determine its strengths and weaknesses, potential opportunities and threats, resources required for implementation, and prospects for ultimate success. There are two criteria to consider when assessing feasibility: required cost and expected value.

1. Technical feasibility

This assessment focuses on the specialized coffers available to the association. It helps associations determine whether specialized coffers meet capacity and whether the specialized platoon is suitable to restate ideas into working systems. Specialized feasibility also includes an evaluation of the tackle, software, and other specialized conditions of the proposed system. As an inflated illustration, the association would not want to try to put Star Trek transporters in their structure the design isn't presently technically doable.

2. Economic Feasibility

In order to help businesses evaluate a project's viability, costs, and advantages before allocating financial resources, this assessment usually entails a cost/benefit analysis of the project. In addition, it acts as an impartial evaluation of the project and raises its credibility, assisting decision-makers in ascertaining the favorable financial advantages the project will bring to the company.

3. Legal Feasibility

This evaluation analyzes whether any perspective of the proposed venture clashes with lawful necessities such as zoning laws, information assurance acts or social media laws. Suppose an organization needs to build a unused office building at a specific area. A possibility ponder may reveal that an organization's perfect area is not zoned for that sort of trade. That organization has spared noteworthy time and exertion basically by realizing that their venture was not feasible from the begin.

4. Operational Feasibility

This evaluation involves conducting a study to analyze and determine whether—and how well—the organization's needs can be met by completing the project. Operational feasibility studies also examine how the project plan meets the requirements identified in the requirements analysis phase of system development.

5. Scheduled Feasibility

This evaluation is the most important one for the project's success; if it isn't finished on time, the extension will ultimately fall short. The organisation estimates how long it will take to complete the project when planning achievability. After examining each of these areas, a possibility analysis aids in identifying any obstacles the proposed project might face, such as:

- Internal imperatives: specialised, innovative, financial, resource-related, etc.
- Internal company restrictions: financial, commerce, display, etc.
- External constraints, such as environment, laws & regulations, coordinations, etc.

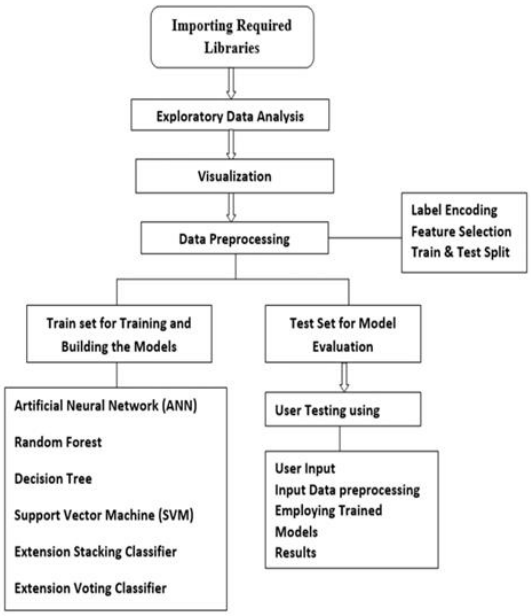
VII. DATA FLOW DIAGRAM

1. A DFD is also called a bubble map. It's a simple graphical formalism that can be used to represent a system in terms of input data to the system, colorful processing with that data, and affair data generated by the system.

2. Data inflow illustration(DFD) is one of the most important modeling tools. It's used to model system factors. These factors are the system process, the data used by the process, the external reality that interacts with the system, and the information flows in the system.

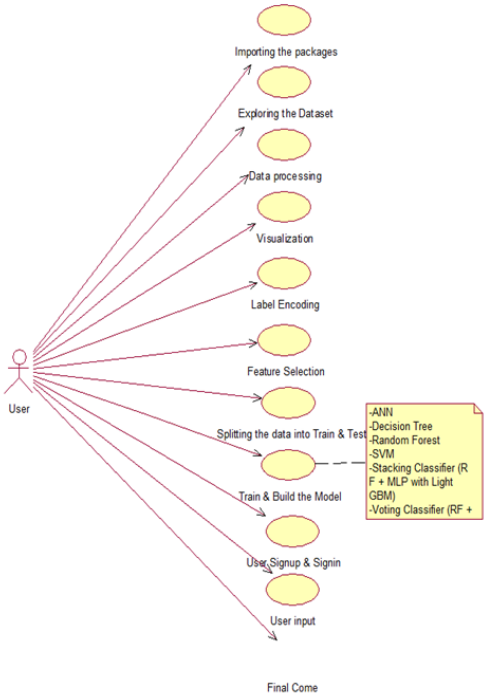
3. DFD shows how information moves through the system and how it's modified by a series of metamorphoses. It's a graphical fashion that shows the inflow of information and the metamorphoses that are applied when moving data from input to affair.

4. DFD is also known as bubble map. DFDs can be used to represent a system at any position of abstraction. DFDs can be divided into situations that represent adding information inflow and functional detail.



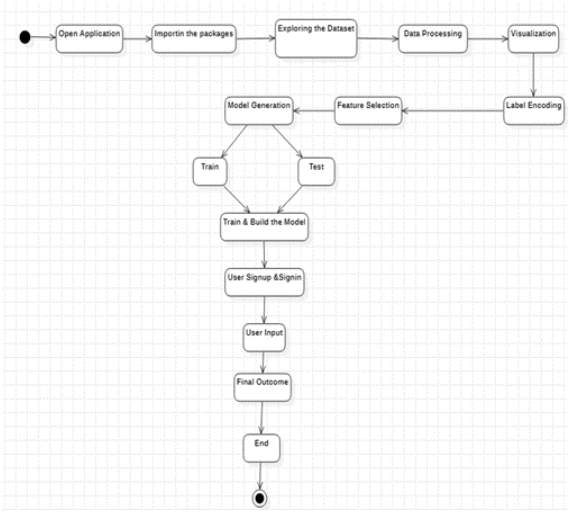
VIII. USE CASE DIAGRAM

A Unified Modeling Language(UML) use case illustration is a type of geste illustration defined and created from use case analysis. Its purpose is to present a graphical overview of the functions handed by the system in terms of actors, their pretensions(represented as use cases) and any dependences between these use cases. The main purpose of a use case illustration is to show which system functions are performed for which actor. The places of the actors in the system can be depicted.



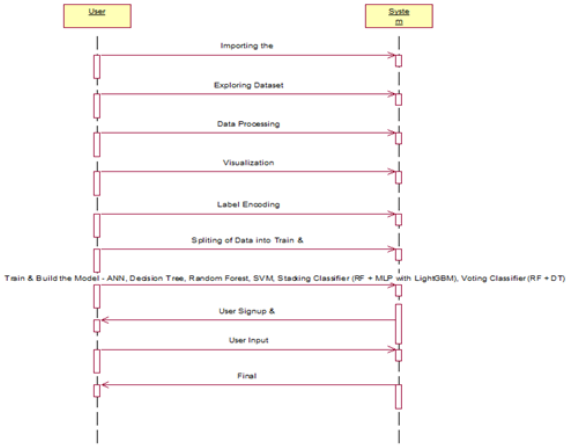
IX. ACTIVITY DIAGRAM

Process flows in the system are captured in an activity diagram. Similar to a state diagram, an activity diagram consists of activities, actions, transitions, initial and final states, and guard conditions.



X. SEQUENCE DIAGRAM

A sequence illustration represents the commerce between colorful objects in a system. An important aspect of a sequence illustration is that it's time-ordered. This means that the exact sequence of relations between objects is shown step by step. Different objects in a sequence illustration interact with each other by passing" dispatches".



XI. IMPLEMENTATION

Data loading:
Using this module we are going to consequence the dataset.

Dataset Description
NSL KDD:

This dataset, which focuses on network infiltration, is the result of KDD'99 Cup duties. The dataset is kept up-to-date and has excellent documentation, along with the Train and test suites contain a manageable amount of records, so it is possible to run experiments on the entire dataset without having to randomly select a subset. As a result, evaluation results from different research projects will be comparable and consistent.

Data Preprocessing: we will use this module to explore the data.

Splitting information into prepare and test: using this module the data will be split into train and test

Model Generation: Model Building – ANN, Decision Tree, SVM, Random Forest, Voting Classifier (RF + DT), Stacking Classifier (RF + MLP with LightGBM). Calculation of accuracy of algorithms

User Signup and Login: Utilizing this module will lead to registration and login.

User Input: Utilizing this module will get input for forecast.

Prediction: Show the last expectation.

XII. ALGORITHMS

ANN: Artificial neural networks(ANNs) or neural networks are computational algorithms. Its end was to pretend the geste of natural systems composed of" neurons". ANNs are computational models inspired by the central nervous system of an beast.

Decision Tree: A decision tree is anon-parametric supervised literacy algorithm that's used for both bracket and retrogression tasks. It has a hierarchical tree structure that consists of a root knot, branches, internal bumps, and splint bumps.

Random Forest: Random Forest is a commonly utilized machine learning calculation trademarked by Leo Breiman and Adele Cutler, which combines the yields of numerous choice trees to arrive at a single result. Its ease of utilize and adaptability have empowered its selection, as it handles both classification and relapse problems.

SVM: SVM is a important supervised algorithm that works stylish on lower datasets but on complex bones . Support Vector Machines, or SVMs for short, can be used for both retrogression and bracket tasks, but generally work stylish in bracket problems.

Voting Classifier (RF + DT): A voting classifier is a machine learning estimator that aggregates the results of each base estimator to make predictions. It trains a variety of base models or estimators. Voting decisions for each estimator output can be merged to form the aggregating criteria.

Stacking Classifier (RF + MLP with LightGBM): A stacking classifier is an gathering strategy where the output from numerous classifiers is passed as input to a meta-classifier for the last classification assignment. A stacking classifier approach can be a exceptionally proficient way to execute a multi-classification issue.

XIII. SOFTWARE ENVIRONMENT

Anaconda software helps you build environments for many different Python versions and package versions. Anaconda is also used to make changes packages in your project environment. Additionally, you can use Anaconda to deploy any project you want with just a few mouse clicks. That's why it's perfect for beginners who want to learn Python.

XIV. SYSTEM TESTING

Framework testing, too alluded to as system-level tests or framework integration testing, is a prepare in which a quality confirmation (QA) group assesses how the different components of an application work together in a total coordinates framework or application. Framework testing confirms that the application performs assignments as outlined. This step, a kind of dark box testing, centers on the usefulness of the application. For case, framework testing can check that each kind of client input produces the expecting yield all through the application.

XV. STATIC TEST

An early stage testing strategy is static testing: it is done without actually running the product under development. Basically, such a table check is necessary to find bugs and problems present in the code itself. Such a review is important in the pre-deployment phase, as it helps to avoid problems caused by code errors and defects in the software structure.

XVI. STRUCTURAL TESTING

Software cannot be tested efficiently unless it is run. White-box testing, another name for structural testing, is necessary to find and correct flaws and faults that surface during the pre-production phase of the software development process. Regression testing is

being used for unit testing depending on the programme structure. To expedite the development process at this point, it is typically an automated procedure operating inside the test automation framework. With complete access to the software's architecture and data flows (data flows testing), developers and quality assurance engineers are able to monitor any alterations (mutation testing) in the behaviour of the system by contrasting the test results with those of earlier iterations (control flow testing).

XVII. BEHAVIORAL TESTING

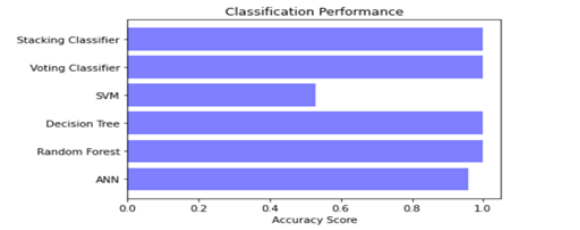
The final phase of testing focuses on the software's reactions to various activities rather than the mechanisms behind these reactions. In other words, behavioral testing, too known as black-box testing, presupposes running various tests, for the most part manual, to see the item from the user's point of see. QA engineers usually have some specific information about the business or other purposes (the 'black box') of the software to run usability tests, for example, software used to conduct and react to bugs as regular users of the product would. Behavioral testing can moreover include computerization (relapse tests) to kill human blunder if monotonous exercises are required. For example, you should write to 100 registration forms on a website to see how the product copes with such activity, so automating this test is better.

XVIII. TEST CASES

S.NO	INPUT	If available	If not available
1	User signup	User get registered into the application	There is no process
2	User signin	User get login into the application	There is no process
3	Enter input for prediction	Prediction result displayed	There is no process

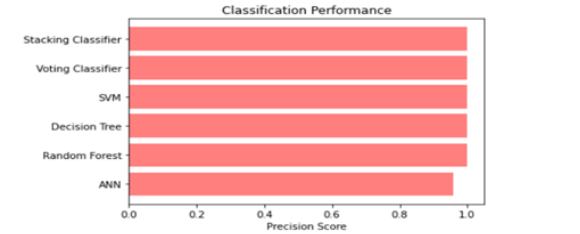
XIX. PERFORMANCE EVALUATION

```
import matplotlib.pyplot as plt2
plt2.barh(y_pos, accuracy, align='center', alpha=0.5,color='blue')
plt2.yticks(y_pos, classifier)
plt2.xlabel('Accuracy Score')
plt2.title('Classification Performance')
plt2.show()
```



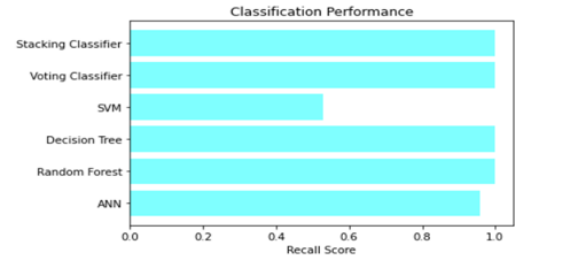
XX. ACCURACY COMPARISON

```
plt2.barh(y_pos, precision, align='center', alpha=0.5,color='red')
plt2.yticks(y_pos, classifier)
plt2.xlabel('Precision Score')
plt2.title('Classification Performance')
plt2.show()
```



XXI. PRECISION COMPARISON GRAPH

```
plt2.barh(y_pos, recall, align='center', alpha=0.5,color='cyan')
plt2.yticks(y_pos, classifier)
plt2.xlabel('Recall Score')
plt2.title('Classification Performance')
plt2.show()
```



If a software build achieves the desired results in system testing, it goes through final review through acceptance testing before going into production where users consume the software. The application development team logs all defects and determines what kinds and amounts of defects are tolerable.

CONCLUSION

In this work, we propose a Network Intrusion Detection System (NIDS) machine learning framework for two-stage hybrid ensemble learning and detection of various known and unknown attacks on high-dimensional network data. The proposed automatic feature selection engine can be used with high-dimensional network databases to identify the most important features needed for multi-level attack classification. In addition, we identify the shortcomings of existing feature selection methods for network data and introduce new feature engineering techniques that improve attack detection. The proposed framework is built using a hybrid of two learning algorithms. Despite the abundance of One-vs-Rest based learning algorithms in the literature, it has become apparent that there is not enough research on One-vs-One ensemble learning algorithms for intrusion detection. Our two-stage architecture, built on a One-vs-One framework, is capable of learning one attack type from another attack type, resulting in multi-level classification accuracy. The proposed framework is tested using two well-referenced databases on wired and wireless networks, and the results are compared with other studies in the literature. The THEAFS-DT model works best for cable applications with good detection rates and low false alarm rates. THE-AFS-RF model is best for wireless applications with good detection rate and low false alarm rate. Wireless applications have surpassed other leading studies in the literature that have tried to build a generalized model that works in both wired and wireless applications.

REFERENCES

[1] F. Obite, E. T. Jaja, G. Ijeomah, and K. I. Jahun, ‘‘The evolution of Ethernet Passive Optical Network (EPON) and future trends,’’ Optik, vol. 167, pp. 103–120, Aug. 2018.

[2] B. Bellalta, L. Bononi, R. Bruno, and A. Kassler, ‘‘Next generation IEEE 802.11 wireless local area networks: Current status, future directions and open challenges,’’ Comput. Commun., vol. 75, pp. 1–25, Feb. 2016.

[3] E. J. Oughton, W. Lehr, K. Katsaros, I. Selinis, D. Bubley, and J. Kusuma, ‘‘Revisiting wireless internet connectivity: 5G vs Wi-Fi 6,’’ Telecommun. Policy, vol. 45, no. 5, Jun. 2021, Art. no. 102127.

[4] S. Hajiheidari, K. Wakil, M. Badri, and N. J. Navimipour, ‘‘Intrusion detection systems in the Internet of Things: A comprehensive investigation,’’ Comput. Netw., vol. 160, pp. 165–191, Sep. 2019.

[5] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, and S. C. De Alvarenga, ‘‘A survey of intrusion detection in Internet of Things,’’ J. Netw. Comput. Appl., vol. 84, pp. 25–37, Apr. 2017.

[6] R. Mitchell and I.-R. Chen, ‘‘A survey of intrusion detection in wireless network applications,’’ Comput. Commun., vol. 42, no. 3, pp. 1–23, Apr. 2014.

[7] W. Wang, S. Jian, Y. Tan, Q. Wu, and C. Huang, ‘‘Representation learningbased network intrusion detection system by capturing explicit and implicit feature interactions,’’ Comput. Secur., vol. 112, Jan. 2022, Art. no. 102537.

[8] M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, ‘‘Deep abstraction and weighted feature selection for Wi-Fi impersonation detection,’’ IEEE Trans. Inf. Forensics Security, vol. 13, no. 3, pp. 621–636, Mar. 2018.

[9] M. H. L. Louk and B. A. Tama, ‘‘Dual-IDS: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system,’’ Exp. Syst. Appl., vol. 213, Mar. 2023, Art. no. 119030.

[10] M. Akbanov, V. G. Vassilakis, and M. D. Logothetis, ‘‘Ransomware detection and mitigation using software-defined networking: The case of WannaCry,’’ Comput. Electr. Eng., vol. 76, pp. 111–121, Jun. 2019.

[11] C. Adams, ‘‘Learning the lessons of WannaCry,’’ Comput. Fraud Secur., vol. 2018, no. 9, pp. 6–9, Jan. 2018.

[12] A. Cooke, K. Renaud, D. Spence, and C. Tankard, ‘‘US authorities recover most of colonial pipeline ransom,’’ Netw. Secur., vol. 2021, no. 6, pp. 1–2, 2021. [Online]. Available: https://www.magonlinelibrary.com/doi/full/10.1016/S1353-4858%2821%2900057-X

[13] J. W. Mikhail, J. M. Fossaceca, and R. Iammartino, ‘‘A semi-boosted nested model with sensitivity-based weighted binarization for multi-domain network intrusion detection,’’ ACM Trans. Intell. Syst. Technol., vol. 10, no. 3, pp. 1–27, May 2019.

[14] Y. Zhou, T. A. Mazzuchi, and S. Sarkani, ‘‘M-AdaBoost-A based ensemble system for network intrusion detection,’’ Exp. Syst. Appl., vol. 162, Dec. 2020, Art. no. 113864.

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

[16] R. Kumar, A. Malik, and V. Ranga, ‘‘An intellectual intrusion detection system using hybrid hunger games search and remora optimization algorithm for IoT wireless networks,’’ Knowl.-Based Syst., vol. 256, Nov. 2022, Art. no. 109762.

[17] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, ‘‘Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset,’’ IEEE Commun. Surveys Tuts., vol. 18, no. 1, pp. 184–208, 1st. Quart., 2016.

[18] B. A. Tama and S. Lim, ‘‘Ensemble learning for intrusion detection systems: A systematic mapping study and cross-benchmark evaluation,’’ Comput. Sci. Rev., vol. 39, Feb. 2021, Art. no. 100357.

[19] G. Folino and P. Sabatino, ‘‘Ensemble based collaborative and distributed intrusion detection systems: A survey,’’ J. Netw. Comput. Appl., vol. 66, pp. 1–16, May 2016.

[20] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, ‘‘Building an efficient intrusion detection system based on feature selection and ensemble classifier,’’ Comput. Netw., vol. 174, Jun. 2020, Art. no. 107247.