

Received 3 May 2024, accepted 16 May 2024, date of publication 24 May 2024, date of current version 18 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3404778

RESEARCH ARTICLE

Securing the IoT Cyber Environment: Enhancing Intrusion Anomaly Detection With Vision Transformers

LARAIB SANA¹, MUHAMMAD MOHSIN NAZIR¹,
JING YANG², (Graduate Student Member, IEEE), LAL HUSSAIN^{3,4},
YEN-LIN CHEN⁵, (Senior Member, IEEE), CHIN SOON KU⁶,
MOHAMMED ALATIYYAH⁷, SULAIMAN ABDULLAH ALATEYAH⁸,
AND LIP YEE POR², (Senior Member, IEEE)

¹Department of Computer Science, Lahore College for Women University, Lahore, Punjab 54000, Pakistan

²Department of Computer System and Technology, Universiti Malaya, Kuala Lumpur 50603, Malaysia

³Department of Computer Science and IT, Neelum Campus, The University of Azad Jammu and Kashmir, Azad Kashmir 13230, Pakistan

⁴Department of Computer Science and IT, King Abdullah Campus, The University of Azad Jammu and Kashmir, Muzaffarabad, Azad Kashmir 13100, Pakistan

⁵Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei 106344, Taiwan

⁶Department of Computer Science, Universiti Tunku Abdul Rahman, Kampar 31900, Malaysia


⁷Department of Computer Science, College of Sciences and Humanities-Aflaj, Prince Sattam Bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia

⁸Department of Computer Engineering, College of Computer, Qassim University, Buraydah 52571, Saudi Arabia

Corresponding authors: Yen-Lin Chen (ylchen@mail.ntut.edu.tw), Chin Soon Ku (kucs@utar.edu.my), and Muhammad Mohsin Nazir (mohsinsage@gmail.com)

This work was supported in part by the National Science and Technology Council in Taiwan under Grant NSTC-112-2221-E-027-088-MY2 and Grant NSTC-112-2622-8-027-008; in part by the Ministry of Education of Taiwan titled “The Study of Artificial Intelligence and Advanced Semiconductor Manufacturing for Female STEM Talent Education and Industry-University Value-Added Cooperation Promotion” under Grant 1122302319; and in part by the Universiti Tunku Abdul Rahman (UTAR) Financial Support for Journal Paper Publication Scheme through UTAR, Malaysia.

ABSTRACT The ever-expanding Internet of Things (IoT) landscape presents a double-edged sword. While it fosters interconnectedness, the vast amount of data generated by IoT devices creates a larger attack surface for cybercriminals. Intrusions in these environments can have severe consequences. To combat this growing threat, robust intrusion detection systems (IDS) are crucial. The data comprised by this attack is multivariate, highly complex, non-stationary, and nonlinear. To extract the complex patterns from this complex data, we require the most robust, optimized tools. Machine learning (ML) and deep learning (DL) have emerged as powerful tools for IDSs, offering high accuracy in detecting and preventing security breaches. This research delves into anomaly detection, a technique that identifies deviations from normal system behavior, potentially indicating attacks. Given the complexity of anomaly data, we explore methods to improve detection performance. This research investigates the design and evaluation of a novel IDS. We leverage and optimize supervised ML methods like tree-based Support Vector Machines (SVM), ensemble methods, and neural networks (NN) alongside the cutting-edge DL approach of long short-term memory (LSTM) and vision transformers (ViT). We optimized the hyperparameters of these algorithms using a robust Bayesian optimization approach. The implemented ML models achieved impressive training accuracy, with Random Forest and Ensemble Bagged Tree surpassing 99.90% of accuracy, an AUC of 1.00, an F1-score, and a balanced Matthews Correlation Coefficient (MCC) of 99.78%. While the initial deep learning LSTM model yielded an accuracy of 99.97%, the proposed ViT architecture significantly boosted performance with 100% of all metrics, along with a validation accuracy of 78.70% and perfect training accuracy. This study demonstrates the power of our new methods for detecting and stopping attacks on Internet of Things (IoT) networks. This improved detection offers a three-pronged approach to security: increased system

The associate editor coordinating the review of this manuscript and approving it for publication was Sangsoon Lim .

reliability through attack prevention, enhanced security by swiftly identifying and mitigating fraudulent activity, and optimized network performance by preventing malicious attacks. Consequently, these methods offer significant potential for fortifying the security of IoT networks.

• **INDEX TERMS** Vision transformers, anomaly detection, intrusion detection, IoT, deep learning.

I. INTRODUCTION

Intrusion detection systems for the IoT aim to identify and respond to potential intrusions or security breaches in real-time. These systems monitor network traffic, device behavior, and communication patterns to detect anomalies, suspicious activities, or known attack patterns. Some common techniques used in IoT intrusion detection include:

1. **Signature-based detection:** This method involves comparing network traffic or device behavior against known attack signatures or patterns. If a match is found, it indicates a potential intrusion.
2. **Anomaly-based detection:** Anomaly-based detection involves establishing a baseline behavior for IoT devices and networks and then identifying any deviations from the normal patterns. Unusual or unexpected behavior can indicate a possible intrusion.
3. **Behavior-based detection:** This technique focuses on monitoring the behavior of individual IoT devices. It establishes expected behavior for each device and flags any deviations that may suggest a compromise or unauthorized access. Machine learning algorithms can be trained to analyze IoT data and identify patterns that indicate malicious activities. These algorithms can learn from historical data and adapt to new threats over time.

Anomaly detection is important in various fields as it helps identify unusual patterns, events, or behaviors that deviate from the norm [1], [2], [3], [4]. It has applications in fields such as finance, cybersecurity, healthcare, manufacturing, and more.

In cybersecurity, anomaly detection helps identify potential threats or attacks on computer systems or networks [5]. By analyzing network traffic and identifying unusual patterns, anomalies can be detected, and security measures can be put in place to prevent or mitigate cyberattacks [5], [6].

In healthcare, anomaly detection is used to identify patients with unusual symptoms or disease progression, allowing for early intervention and treatment [1]. It can also be used to detect outbreaks of infectious diseases or other public health threats.

As our world is known as the global world, more than 70% of our world population is using active internet connections on multiple digital devices. People are taking advantages and opportunities from the internet, but many cyber attackers are using these internet services illegally and trying to misuse them by generating many cyberattacks. They use different cyberattacks to steal people's information and hijack internet services.

To handle such cyberattacks and threats, researchers have proposed many intrusion detection systems using machine

learning and deep learning techniques. The complexity and evolving nature of IoT systems, along with the exponential growth of IoT network traffic data, pose challenges for existing anomaly detection approaches to effectively detect and defend against attacks, particularly unknown attacks.

To address this issue, this paper presents a novel and enhanced anomaly detection approach based on machine learning for IoT networks. The proposed approach aims to detect various anomalies, specifically abnormal network traffic, in a highly contextual and scalable IoT network infrastructure. By employing this approach, it becomes possible to accurately identify the behavior of normal and abnormal traffic with high detection accuracy and a low false-positive rate. Ultimately, this approach enhances the security of both sensitive data and IoT devices, thereby maximizing overall security in IoT environments.

The significant contributions of this paper can be summarized as follows:

- **Introduction to Vision Transformers (ViT) for Anomaly Detection:** The paper pioneers the use of Vision Transformers in the field of anomaly detection, expanding the application of ViT models beyond image classification to effectively identify anomalies in various types of data, including network traffic.
- **Application in Intrusion Detection:** By applying ViT models to intrusion detection systems, the paper contributes to enhancing network security. It introduces a novel approach to improving the accuracy and effectiveness of detecting cyber threats, a critical aspect of safeguarding computer networks.
- **Benchmarking and Comparative Analysis:** The paper evaluates ViT-based anomaly detection on the NSL-KDD dataset and conducts a comparative analysis with traditional machine learning algorithms. This empirical assessment demonstrates the potential advantages of ViT models and provides valuable insights into their strengths in handling complex data patterns.

The structure of the remaining paper is outlined as follows: Section II provides an overview of existing research on anomaly detection in IoT networks and presents a taxonomy of existing datasets for anomaly and intrusion detection. Section III defines and introduces the proposed anomaly detection method specifically designed for IoT networks. Section IV presents the experimental results and discusses the findings obtained from applying the proposed method. Section V concludes the paper by summarizing the work presented and potentially highlighting future directions for research in this area.

II. RELATED WORK

The internet has become an indispensable part of our lives, and its reach continues to grow. According to the Cisco Annual Internet Report [7], the number of global internet users is expected to surge from 3.9 billion in 2018 to 5.3 billion by 2023. This growth is further amplified by the IoT. The IoT seamlessly integrates various physical devices, like smart bulbs, refrigerators, and thermostats, into a connected network. These devices communicate using technologies like Bluetooth Low Energy (BLE), WiFi, and ZigBee [8], [9], [10], [11]. The applications of IoT extend far beyond smart homes, encompassing smart transportation, industrial automation, agriculture, and healthcare [12]. This interconnectedness, however, creates a vulnerability to cyberattacks. The very nature of being connected exposes the IoT ecosystem to various threats at different layers of its architecture. Makhdoom et al. [13] provide a detailed overview of these attacks, categorized based on malware types. Similarly, surveys by [14] and Zarpelão et al. [15] highlight the prevalence of IoT-enabled cyberattacks and the importance of IDS in mitigating them.

Researchers have categorized IDS for IoT networks based on factors like placement within the network, detection methods employed, targeted security threats, and validation strategies used [16]. Zargar et al. [16] specifically explored distributed denial-of-service (DDoS) attacks on IoT networks and classified various countermeasures. An IDS safeguards communication between devices and detects intrusions across different IoT network layers. Numerous IDS solutions have been developed to secure internet communication [17], [18], [19], [20]. These systems actively monitor network activity for malicious behavior and send alerts to system administrators upon detecting attacks. However, IoT devices often face limitations due to their small size and low battery capacity. This translates to limited computational power. Additionally, they rely on lightweight communication protocols. Consequently, any attack detection algorithms designed for IoT networks must be lightweight and energy-efficient.

Several factors make IoT devices inherently vulnerable: limited computational power, variations in equipment, software, and communication protocols [21], [22], [23]. This creates a significant gap between the security demands placed on these devices and their actual defense capabilities [24]. Specifically, IoT devices with limited resources—processing power, memory, and battery life—struggle to handle computationally intensive security tasks that generate significant processing and communication overhead [24]. Implementing complex and robust security protocols on such devices is simply not feasible. The heterogeneity of the IoT landscape further complicates the development and deployment of security solutions. Striking a balance between effective security and efficient performance becomes a major challenge, considering the vast diversity of devices and their capabilities.

The rise of machine learning techniques has brought promise to securing IoT networks. Algorithms like KNN,

Random Forest, and Naive Bayes offer good performance with low complexity and reasonable computation time for resource-constrained devices. Anthi et al. [25] proposed a three-layer IDS for smart home IoT devices. They conducted a testbed experiment with eight devices, monitored network traffic, and simulated various attacks. The captured data was categorized into four main attack types: denial-of-service (DoS), man-in-the-middle, reconnaissance, and replay. Their system utilizes three layers:

- **Device Identification:** Scans the network to identify connected IoT devices.
- **Packet Classification:** Classifies network packets as malicious or normal.
- **Attack Type Identification:** It uses nine ML classifiers to categorize malicious packets into four attack types and selects the best-performing one.

However, their work lacks a clear definition of the data features used for classification. Other studies explore different ML approaches for intrusion detection. Wenjuan Li et al. [26] proposed a mechanism that utilizes both labeled and unlabeled data for training, but their evaluation relied on outdated DARPA (KDD99) data. Horng et al. [27] used clustering to select relevant features for each attack type from the KDD Cup 99 dataset. They then employed separate SVM classifiers for each attack type, exploring both individual and combined configurations. Eesa et al. [28] introduced a novel feature separation technique based on a cuttlefish algorithm, followed by a decision tree classifier for intrusion detection. Challenges remain in building intelligent data mining systems for intrusion detection due to the large datasets involved in the training phase. The article by [29] proposes a feature selection algorithm (“Highest Wins”) to address this challenge by identifying the most relevant features from training datasets. Li et al. [30] presented a two-step AI-based intrusion detection system using a software-defined approach. Their system leverages a weighted voting system with a Random Forest algorithm to categorize network flows.

Anomaly detection in multivariate time series data is crucial for various applications, from network intrusion detection in cybersecurity to identifying faulty equipment in industrial settings [31], [32], [33], [34], [35]. Unlike univariate time series that analyze a single variable over time, multivariate time series deal with multiple interrelated variables [36], [37], [38], [39], [40], [41], [42], providing a richer picture of the underlying processes [43], [44], [45], [46], [47]. This expanded view allows for more sophisticated anomaly detection techniques.

Recently, researchers utilized methods to detect anomalies from different categories, such as statistical models, which rely on pre-defined assumptions about the underlying distribution of the data [48], [49]. Popular examples include ARIMA (Autoregressive Integrated Moving Average) and Hidden Markov Models (HMMs). While offering interpretability, they struggle with complex or unknown data distributions. ML approaches learn patterns from historical

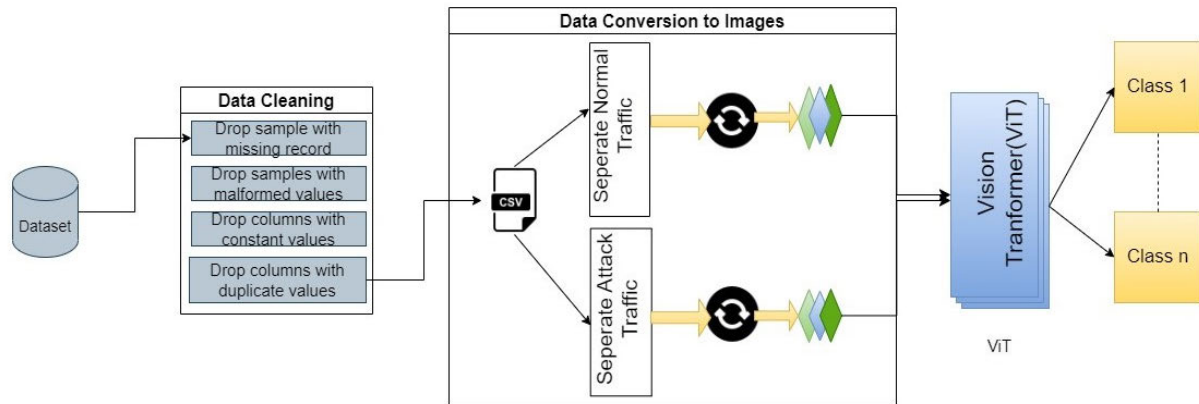


FIGURE 1. Proposed methodology.

data to identify anomalies. Techniques like KNN, SVMs, and Random Forests excel at capturing complex relationships in the data. However, they can be susceptible to high dimensionality (too many variables), noise, and asynchronous anomalies (anomalies appearing at different times in different variables). DL architectures like RNNs, LSTM networks, and vision transformers can handle complex data patterns and temporal dependencies. They learn intricate relationships between multiple variables, making them powerful for anomaly detection. However, deep learning models often require large amounts of data for training and can be challenging to interpret.

The ability to detect anomalies in multivariate time series data is critical across a wide range of fields [50], [51], [52]. From pinpointing fraudulent transactions in finance to identifying precursor signals for equipment failure in industrial settings, these techniques offer significant practical value [53], [54]. Unlike univariate analysis with its single-variable focus, multivariate time series analysis incorporates the interplay between multiple variables, leading to a more comprehensive understanding of the underlying system. This richer data perspective allows for the development of sophisticated anomaly detection methods.

Machine learning is widely used to create efficient intrusion detection systems that can accurately distinguish between normal and malicious patterns. An efficient detection model ensures quick and accurate identification of potential intrusions.

One study used SVM to design intrusion detection [55], while another study used decision trees to identify malicious and benign traffic. The main aim is to improve network security by designing effective and safe intrusion detection systems. Despite their effectiveness in many domains, these approaches are not capable of detecting network attacks effectively due to the challenges posed by the large volume of data with complex feature representations. So, authors started to use neural networks for designing such intrusion detection systems.

Deep learning has provided promising results for large datasets, which has attracted many authors to employ deep learning techniques for designing intrusion detection systems. The author has used a convolutional neural network (CNN) to propose an intrusion detection system [56], [57]. In another study, the Recurrent Neural Network (RNN) was used to propose an intrusion detection system [58]. Therefore, with the passage of time, RNN and CNN started to become famous for handling such types of problems. However, many deep learning-based approaches to intrusion detection have performed well for attack detection. But these techniques have some shortcomings that need to be addressed. Training deep learning models like RNN and LSTM is costly, and their neural networks are excessively deep. RNN is also a sequential model, which does not work better for intrusion detection. While CNNs are capable of parallel computation and can capture long-range features with sufficient depth, studies have demonstrated that deep convolutional models are not as effective. This is because capturing remote dependencies in CNNs is primarily dependent on the length of paths traveled by forward-backward signals in the network, and the deeper the level of the network, the more difficult it is to obtain these dependencies.

Nowadays, the transformer technique is becoming popular for natural language processing (NLP) and computer vision problems. The transformer technique is based on a self-attention mechanism. It is proven that modern deep learning techniques have provided better performance for image-based problems. But these techniques cannot be used for intrusion detection, as no image dataset is provided for intrusion detection.

In [59], authors have used the concept of vision-based deep learning to design network intrusion detection systems. They have used different image transformers to convert the datasets into images and combine them into color images using three-channel RGB. kPCA (polynomial), kPCA (cosine), kPCA (rbf), PCA with the polynomial kernel, i.e., and PCA with the sigmoid kernel, i.e., kPCA (sigmoid).

And kPCA (polynomial) are used for image transformation. Their model outperforms compared to prior grayscale-based images, although processing the color images requires high processing and memory resources, which makes this model unsuitable for high network traffic in real-time environments.

The authors have designed another intrusion detection model using improved ViT to solve the issue of memory in recurrent neural networks [60]. Their proposed model worked faster than RNN by using the concept of parallelization. Instead of direct segmentation, zero-padding and sliding window mechanisms have been used. They used an NSL-KDD cup for their experiments. To handle the outlier and class imbalance problem, the HFocal loss function has been used. The model provided better accuracy compared to other models, but its running time is comparatively higher than that of another ML model [60].

The main challenges for designing a robust and efficient IDS are unpredictable and unknown attacks, imbalances, and high-dimensional datasets. In [61], the authors proposed the IDS, in which a variant transformer based on an additional multi-head self-attention layer has been designed for feature selection and feature reduction. Synthetic Minority Over-sampling Technique (SMOTE) had been used to handle the imbalanced class samples. The proposed transformer helped to solve the issue of overfitting. The CICIDS 2017 and CIC-DDoS 2019 datasets have been used to evaluate the proposed model. Their proposed model outperforms the other models by providing 98.4% accuracy [61].

The authors have proposed the IDS model using hybrid convolutional neural networks to handle sensitive attacks. They have used LSTM to extract the important features from the datasets and perform comparisons against the original set of features. After extracting the important features, the model was trained using CNN. The proposed model provided a better result with 98% accuracy compared to other ML and DL models. However, they have not explained their architecture in detail [62].

In another study, authors proposed the IDS based on a semi-supervised deep learning approach [63]. They used both labeled and unlabeled IoT data traffic. Their study mainly focused on the issue of the learning of spatiotemporal representation. To handle this issue, a module named multi-scale residual temporal convolution was designed. They have used an attention model to identify important information to increase the efficiency of training. Their model achieved 99.69% accuracy for attack detection [63]. However, it was not clear how their model would work with huge network traffic and a variety of unknown IoT attacks.

Another piece of research has focused on the use of ensemble models in the domain of intrusion detection [64]. Seven machine learning classifiers, such as gradient-boosted machines, random forests, extremely randomized trees, multi-layer perceptrons, and regression trees, have been used for the detection of DDoS attacks. They also performed simulations to check the response time of each classifier.

To analyze the differences among the classifiers, the author used the Nemenyi and Friedman statistical tests. Their results concluded that extreme gradient-boosting classifiers, regression trees, and classification are suitable for the design of IDS [64]. However, they have not discussed the response of each classifier to the routing attacks.

In another study, the author presented a hierarchical hybrid approach for IoT security. They have used a multimodal deep autoencoder for the detection of anomalies and a soft-output classifier for attack detection [65]. BotIoT was used to evaluate the proposed model. Their model worked better in the detection of unknown attacks, especially for DDoS and DoS. They claimed that their model is lightweight and works best for on-device implementation in a resource-constrained IoT environment.

Murder et al. have used multi-layered recurrent neural networks to design an intelligent intrusion detection system [58]. Their proposed framework consists of two steps for traffic analysis and attack detection. They used two different deep recursive neural networks with different hyperparameters and internal structures. The first layer of the neural network identified the attacks. They used an updated version of the backpropagation algorithm for training, which improved the effectiveness of attack detection. Since DoS attacks are one of the main attack types that affect IoT systems, in addition to being detected, the first layer shows attack detection for DoS attacks. Using a second-layer network that has a different recursive gain, adjustable parameters, and internal structure, the first filter's normal response is refiltered for high security. The proposed model worked better than other models in terms of training time [58].

In another study, the author used the concepts of deep learning, transfer learning, and hyperparameter optimization to design an intrusion detection system [66]. They converted the datasets into images using chunk-based transformation and used ensemble learning to improve the performance of CNN. The CICIDS 2017 dataset and car hacking datasets were used to evaluate the model. Their model worked better and provided a higher F1 score compared to another proposed model in this domain [66].

Another study has also used the concepts of transfer learning and vision transformers for network intrusion detection systems [67]. They proposed a preprocessing technique to convert the features into four-channel images. They used those images for classification to train and test the pre-trained deep learning model. BOUN DDoS datasets and UNSW-NB15 were used to evaluate the proposed technique. Results showed that their technique improved detection accuracy [67].

In the majority of the IDS datasets, the distribution of attack classes such as R2L and U2R is imbalanced, which can impact the performance of IDS [68]. To handle this issue, Yuelei Xiao et al. proposed a network intrusion detection system using a simplified residual network (ResNet). They used the parametric rectified linear unit (PReLU) function instead

of RELU. Results showed that their proposed model provided better accuracy and recall in attack detection compared to state-of-the-art IDS [68].

The authors in another study emphasize the importance of CNN compared to a sequential deep learning model. A model based on EfficientNet and ResNet 50 has been designed by Martin Kody et al.; they deployed two different CNN algorithms on the TON IoT dataset to differentiate their performance [69]. Results showed that both algorithms worked better in terms of recall compared to prior-designed IDS. The authors have also used ResNet for network intrusion detection systems, but they did not use any transformation techniques to convert the network traffic into images. However, the proposed model provided 99.9% accuracy in DDoS and DoS attack detection [70].

In 2020, Jiarui Song et al. proposed a neural network based on attention mechanisms (LSTM and ResNet) to extract the important patterns of network traffic. They then preprocessed the identified patterns using the proposed feature selection mechanism based on light GBM [71]. In the end, they trained their proposed model using those features. Their proposed model outperforms compared to other proposed IDS [71].

In [72], the author uses the concept of a transformer for network intrusion detection systems. They proposed the model using ResNet 50 and CNN. They evaluate their model against three well-known IDS datasets: UNSWNB 15, NSL-KDD, and CICIDS 2017. Their model provided more than 90% accuracy compared to existing ML algorithms. It was concluded that the visual transformer has great potential for improving the performance of IDS [72].

Tala et al. [73] have also used Resnet 50 to propose an IDS system for smart grids. They used the technique of deep insight to convert the numerical datasets into images. UNSW datasets have been used for evaluating the model. Results showed that their proposed model provided better accuracy in the detection of denial-of-service attacks [73]. However, they did not discuss the detection of other types of attacks.

Om Kumar et al. [74] have proposed the IDS using recurrent kernel CNN and Modified Monarch Butterfly Optimization. They used the min-max technique for pre-processing and improved battle royale optimization for the extraction of optimal features. The CICIDS-2017 and N-BaIoT datasets have been used for the evaluation of the model. Results showed that modified monarch butterfly optimization improved the performance of the DL classifier. The proposed model achieved 99.5% accuracy for CICIDS-2017 and 99.96% accuracy for the N-BaIoT dataset [74].

Recently, Khan H. and co-workers [75], [76], [77], [78], [79] developed unsupervised methods from deep learning, autoencoder-based Gaussian mixture models, and generative adversarial networks to detect anomalies in social networks. In the fight against cyberattacks, intrusion detection systems (NIDS) are gaining a significant boost from transformer-based models. These models offer improved accuracy and efficiency in threat detection, particularly for novel and

unseen threats. Unlike traditional methods, transformers allow NIDS to learn from and adapt to past data, making them a powerful tool.

Transformers are a deep learning architecture originally designed for NLP tasks like translation and text classification [80]. However, their success has extended to other domains, including network traffic analysis and classification [81], image recognition [82], and intrusion detection. This is due to their ability for parallel processing, transfer learning, and achieving high accuracy.

Recently, He et al. [83] utilized deformable ViT based on feature fusion to detect the intrusion. Moreover, Nizam et al. [84] employed CNN-based models to detect the real-time anomaly in multivariate data from industrial IoT.

Several studies have explored transformer integration with NIDS. Wang et al. [85] combined transformers with CNNs to effectively detect DDoS attacks in Software-Defined Networking (SDN) environments. Similarly, Wu et al. [61] proposed an all-in-one intrusion detection solution using transformers. This solution leverages self-attention mechanisms to identify abnormal network activity and classify traffic patterns.

Unveiling the Heart of Transformers: The Self-Attention Layer. This layer allows them to dynamically learn from the context of the input data. It works by calculating three vectors for each element in the input sequence: query, key, and value. These vectors are used to compute a weighted sum of the value vectors. The weights are determined by comparing the query vector to each key vector using a dot product and then applying a SoftMax function. The resulting weighted sum represents the output of the self-attention layer.

In 2021, the author [86] proposed an intrusion detection system based on a bidirectional LSTM. UNSW and the BOT-IOT dataset were used for evaluation purposes. The proposed model is compared with other existing machine and deep learning models. Results showed that their model provided more than 99% accuracy [86].

The aim of this study was to improve the intrusion detection performance of the NSL-KDD dataset, which contains network traffic data with various types of attacks, by utilizing and optimizing the machine learning and deep learning vision transformers.

Anomaly detection in network traffic data is paramount for cybersecurity. Multivariate time series analysis allows for the identification of subtle deviations from normal patterns, potentially revealing cyberattacks or intrusions [87], [88], [89]. In industrial plants, analyzing sensor data from machinery can help predict equipment failures before they occur [90], [91], [92], [93]. By identifying anomalous readings in temperature, vibration, or power consumption data, preventative maintenance actions can be taken, minimizing downtime and maximizing operational efficiency. Financial institutions can leverage multivariate time series analysis to detect fraudulent transactions. By analyzing customer data encompassing purchase location, time, and type alongside

historical spending patterns, anomalies indicative of potential fraud attempts can be flagged [94], [95]. In healthcare, multivariate time series analysis can be applied to analyze patient data like vital signs and lab results. Identifying abnormal deviations from baseline readings can lead to earlier diagnoses and potentially improve patient outcomes [96], [97], [98], [99].

Anomaly detection in multivariate time series data presents a significant challenge. This data, often collected from diverse sources, can be highly complex, non-stationary (meaning its statistical properties change over time), and non-linear. Traditional machine learning algorithms struggle to capture the intricate, non-linear relationships hidden within these complex time series. Recent research has explored machine learning and deep learning methods, but often with default parameters. Hyperparameter optimization offers a powerful approach to further improve anomaly detection performance. This work proposes and utilizes Bayesian optimization, which identifies the most optimal hyperparameters for anomaly detection algorithms. Additionally, ensemble methods in machine learning have proven effective in capturing the complex dynamics of multivariate time series. However, RNNs, a common deep learning method, have limitations in storing long-term correlations within the data. LSTM networks with a bi-directional approach can effectively address this limitation. Furthermore, the Vision Transformer with optimized parameters offers a robust solution. Compared to other deep learning models, it demonstrates a better trade-off between time and memory consumption. This makes it more feasible for implementation on edge devices with limited resources.

In the realm of deep learning, various architectures have been explored for anomaly detection tasks, each with its advantages and limitations. One notable architecture gaining attention is the Vision Transformer (ViT). Originally proposed for image classification tasks, ViT operates on image patches, leveraging self-attention mechanisms to capture global dependencies.

This unique approach has shown promise in handling sequential data beyond images, making it a suitable candidate for analyzing multivariate time series data, such as network traffic or sensor readings.

Compared to traditional deep learning architectures like recurrent neural networks (RNNs) or long short-term memory (LSTM) networks, ViT offers several advantages. Firstly, its self-attention mechanism allows it to capture long-range dependencies more effectively, which is crucial for detecting anomalies in time series data where distant events may influence each other. Secondly, ViT's architecture enables parallel processing of patches, making it computationally efficient, which is desirable for deployment on resource-constrained IoT devices. Lastly, ViT has demonstrated competitive performance while maintaining interpretability, which is often a concern with more complex deep learning models.

By incorporating ViT into the discussion of deep learning techniques for anomaly detection, this response aims to highlight its potential and its comparative advantages in addressing the challenges posed by multivariate time series data analysis in IoT networks.

Our Contributions:

1. **Leveraging Optimized Models:** We employed and optimized robust machine learning and deep learning models to enhance anomaly detection in multivariate time series data.
2. **Ensemble Learning for Improved Prediction:** We incorporated ensemble methods to improve prediction accuracy.
3. **Hyperparameter Optimization for Enhanced Detection:** We optimized the hyperparameters of machine learning and deep learning algorithms using Bayesian optimization, leading to more optimal parameters for improved anomaly detection.
4. **Comprehensive Performance Evaluation:** We evaluated the performance of our methods using comprehensive measures such as the Matthews Correlation Coefficient (MCC) and F-measures. These measures ensure the reliability of the methods by considering the balance between correctly identified positive and negative cases.

III. MATERIALS AND METHODS

Fig. 1 illustrates the overall architecture of the proposed framework, with detailed implementation discussed in Section V. The proposed architecture encompasses several phases: data cleaning, data conversion to images, and the implementation of the vision transformer.

A. DATASET DESCRIPTION

The NSL-KDD dataset comprises network traffic data encompassing different types of attacks and normal traffic. It encompasses a total of 41 features, including fundamental attributes like duration and protocol type, as well as more intricate characteristics such as the number of failed login attempts and the count of root shell prompts. The dataset encompasses five types of attacks: denial of service (DoS), probing, user-to-root (U2R), remote-to-local (R2L), and miscellaneous. These attacks are further divided into subtypes, resulting in a total of 23 attack subtypes.

The NSL-KDD dataset is available in two primary versions: the original and the reduced version. The original dataset comprises over 4 million instances, which can be excessively large for most intrusion detection systems to handle efficiently. In contrast, the reduced version of the dataset is a smaller, more manageable subset, containing approximately 125,000 instances.

The NSL-KDD dataset, containing around 4.9 million records, is imbalanced, with a majority of attack samples (78%) compared to normal ones (16%). The NSL-KDD dataset has become a standard benchmark dataset for

TABLE 1. Types of features.

Feature #	Feature type
1-9	Basic Feature
10-22	Content related feature
23-31	Time related traffic feature
32-41	Host based Traffic feature

evaluating intrusion detection systems. Numerous machine learning and deep learning techniques have been applied to this dataset, including decision trees, support vector machines, random forests, and neural networks. Furthermore, the dataset has served as a basis for assessing the effectiveness of various feature selection and dimensionality reduction techniques in the realm of intrusion detection.

The NSL-KDD dataset is a revised version of the KDD Cup 99 dataset, with redundant records removed to enhance classifier accuracy. The train and test datasets contain a reasonable number of records, enabling experiments to be conducted effectively. The dataset comprises a total of 42 attributes, with attributes 1 to 41 encompassing various traffic flow features. Attribute 42, known as the label, includes one normal class and four attack classes. The 41 features can be categorized into four types, as outlined in Table 1.

B. TYPES OF ATTACK CLASSES

Within the NSL-KDD Cup dataset, there are four distinct attack classes:

Probing: Probing attacks are typically identified using features like source bytes and connection duration. These attacks involve an attacker attempting to gather information about a remote system or network. The primary goal of probing attacks is to gain knowledge about the targeted system or network, including details such as its IP address, operating system, and services.

Denial-of-Service (DoS) Attacks: SYN flooding attacks, a type of DoS attack, can be detected based on features such as the percentage of packets with errors and the number of source bytes. SYN flooding attacks involve overwhelming a victim server by sending an excessive number of SYN requests, preventing it from responding to legitimate requests.

R2L (Remote-to-Local) Attacks: R2L attacks involve an attacker attempting to gain unauthorized local access to a system from a remote machine. These attacks exploit vulnerabilities in the target system's operating system, applications, or network infrastructure. Features associated with R2L attacks may include the number of shell prompts invoked and the number of file creations.

U2R (User-to-Root) Attacks: Features such as the number of shell prompts invoked and the number of file creations are linked to U2R attacks. U2R attacks involve an attacker's attempt to gain root or administrator access to a system by

logging in with a regular account and exploiting security vulnerabilities.

The dataset comprises three files, with the KDD Train+20% subset containing a total of 25,192 records. Additional class divisions and details are presented in Fig. 2.

Table 2 provides an overview of the attack types and their respective classes. Each attack class is further subdivided into additional attack classes for detailed categorization.

C. VISION TRANSFORMER (ViT)

The Vision Transformer represents a specialized adaptation of the Transformer architecture. While the traditional transformer architecture encompasses both an encoder and a decoder, ViT exclusively employs the encoder component. In doing so, it departs from the original Transformer model's design by replacing the standard self-attention layers with specialized self-attention layers that are tailor-made for processing image patches. An additional characteristic of ViT is its frequent utilization of pretraining on extensive image datasets through self-supervised learning. This preliminary training equips the model with valuable visual representations, which can subsequently be fine-tuned for specific supervised tasks such as image classification or object detection.

At the core of the Transformer architecture, and by extension, ViT, lies the self-attention mechanism. This mechanism serves as a foundational element responsible for capturing relationships between distinct words or tokens within a sequence. It empowers the model with the capability to assign varying degrees of importance to each token concerning its counterparts within the sequence. This dynamic allows the model to focus its attention on pertinent information and effectively capture dependencies that extend over extended ranges.

1) SELF-ATTENTION MECHANISM

The self-attention mechanism is a fundamental component of the Transformer architecture, enabling the model to capture relationships among various tokens within a sequence. It operates by dividing the input vector into three distinct vectors: value, key, and query. The output is then computed using a weighted function, with the compatibility between the query and relevant key dictating the weight assigned to each corresponding value.

To calculate these weights for the values in the self-attention mechanism, the input comprises the query along with the keys from the dk vector and the values from the dv vector. The weight assigned to each value is determined through a process involving the computation of the dot product between the query and each key, followed by division by the square root of the key vector's dimension, and finally application of the SoftMax algorithm.

In practice, there are often multiple queries simultaneously, and these queries are consolidated within the Q matrix, while the values and keys are grouped in matrices K and V.

TABLE 2. Attack classes and their types.

Attack Class	Types of Attacks
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint (6)
DoS	Back, Land, Neptune, Pod,Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm
U2R	Sqlattack, Xterm, Ps Buffer_overflow, Loadmodule, Rootkit, Perl
R2L	Guess_Password, Ftp_write, Imap, Phf,Multihop, Warezmaster, Warezclient, Spy,Xlock, Xsnoop, Smpguess, Smpgetattack, Httpunnel, Sendmail, Named (16)

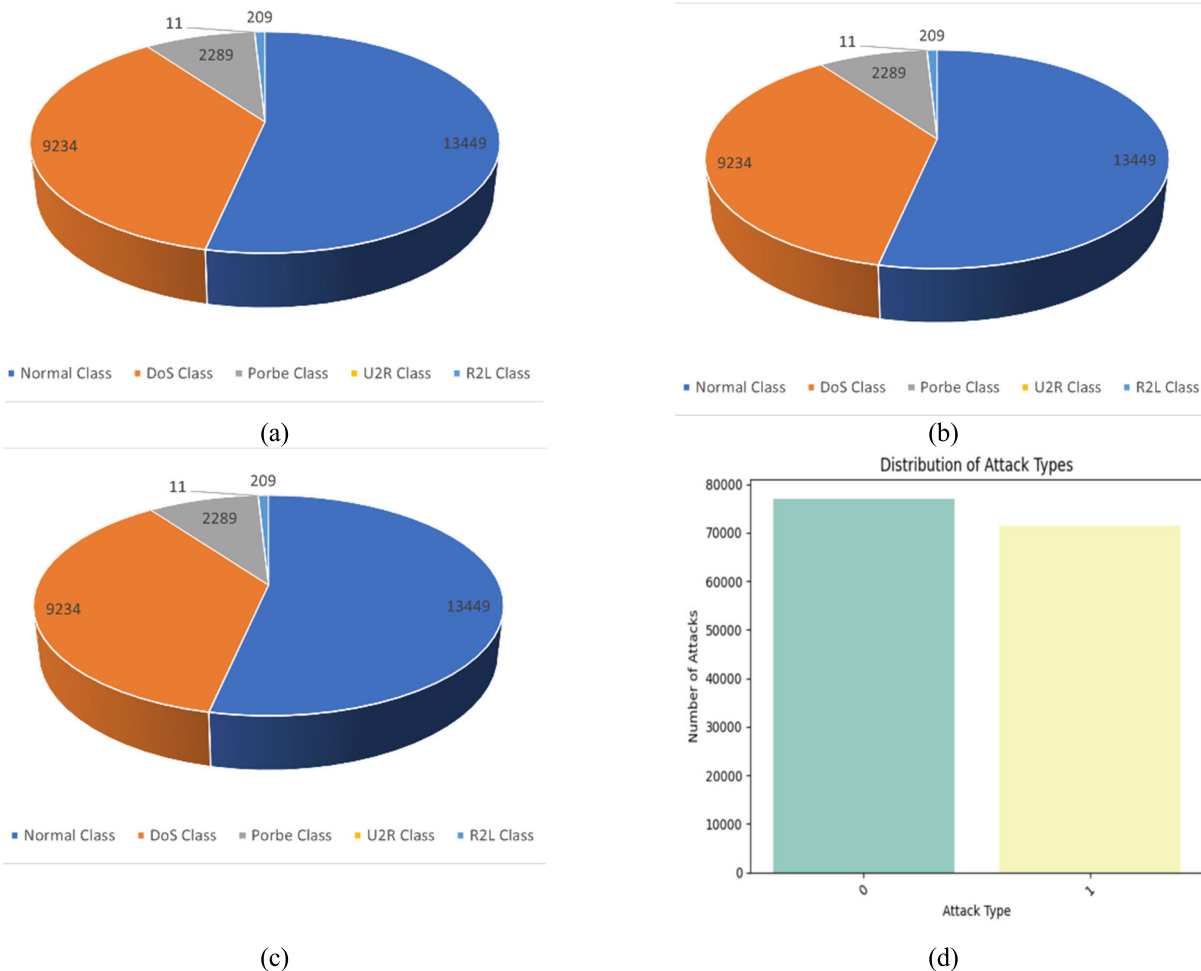


FIGURE 2. Details of attack and normal class in a) KDD train+20, b) KDD Train+, c) KDD Train+, and d) Class distribution in the KDD train (normal: 0, attack: 1).

The combined output is calculated using the formula:

$$Attention(Q, K, V) = Softmax\left(\frac{Q \cdot K^T}{\sqrt{D_K}}\right) \quad (1)$$

Initially, a score is computed for each input vector to gauge the level of attention it should receive, signifying which parts of the input warrant heightened focus. Subsequently, the score values are divided by the dimension of the key vector, followed by the application of the SoftMax function to normalize them within the range of 0 to 1. These normalized scores then translate into probabilities, and the summation of these probabilities is used to weight each value vector.

Additionally, for enhanced training, the scores are further normalized.

Both the encoder and decoder modules within the self-attention layer share identical architectures. The decoder module mirrors the structure of the encoder module. However, a key distinction lies in the origin of the Q matrix, which is derived from the preceding layer in the decoder, while the key and value matrices are derived from the encoder module.

2) MULTI HEAD ATTENTION

Multi-head attention is a deep learning technique commonly employed in transformer-based models to augment the

effectiveness of the self-attention mechanism. This mechanism empowers the model to assess the importance of different segments within the input sequence as it processes it. It allows the model to simultaneously consider multiple facets of the input sequence. Instead of utilizing a single set of query, key, and value matrices, multiple sets of these matrices, each associated with a distinct “head” of attention, are utilized. Each head can focus on different portions of the input sequence and capture diverse features, thereby providing the model with a richer set of information to work with. In multi-head attention, the input sequence is first transformed into query, key, and value vectors. Subsequently, each head conducts its own attention operation on these vectors, yielding multiple sets of weighted values. These sets of values are then concatenated and projected into a final output vector. The advantages of employing multi-head attention encompass improved model performance, as it enables the model to capture more intricate relationships among different segments of the input sequence. Furthermore, it allows the model to direct its attention to various aspects of the input sequence, thereby enhancing its ability to handle data variations effectively. In summary, multi-head attention serves as a potent tool for enhancing the performance of self-attention mechanisms within deep learning models.

3) FEED-FORWARD NEURAL NETWORK (FFN)

The feed-forward neural network comprises two linear transformation layers followed by a nonlinear activation function. Typically, it is applied after the self-attention layer in the architecture. Mathematically, the FFN can be represented as:

$$\text{Feed-forward neural network} = W_2\sigma(W_1X) \quad (2)$$

where X represents the input, W_1 and W_2 are the weight matrices of the first and second layers, respectively, and σ denotes the activation function. The network's output is obtained by passing the weighted sum of the input through the activation function and another layer of weights.

4) RESIDUAL CONNECTION

A residual connection is integrated into every sub-layer of both the encoder and decoder to enhance information flow and overall performance. Subsequently, layer normalization is applied after the residual connection. The outcome of these processes can be expressed as $\text{LayerNorm}(X + \text{Attention}(X))$, where X denotes the input and Attention represents the attention mechanism. In this context, X signifies the input. An alternative variant known as pre-layer normalization has gained popularity, involving the insertion of the normalization layer within the residual connection before the feed-forward neural network and multi-head attention [100].

It's worth noting that batch normalization is not an ideal choice for normalizing transformers, as it can lead to abrupt changes in feature values [101], [102]. Consequently, other normalization methods have been proposed to enhance the training of transformers.

In the final layer of the decoder, the stack of vectors is transformed back into words. This process involves using a linear layer, followed by a SoftMax layer. The linear layer maps the vector to a logit vector with a dimension equivalent to the total number of words in the dictionary, often referred to as d_{word} . To convert the logit vector into probabilities, a SoftMax layer is employed.

The original transformer encoder module has served as a feature extractor for numerous computer vision problems. Transformers possess the capability to extract global information efficiently. Transformers offer advantages over RNN models, mainly due to their parallel processing nature facilitated by fully connected layers and the self-attention mechanism, while RNN models operate sequentially [103].

Therefore, transformers offer substantial benefits for addressing challenges in computer vision and NLP problems.

5) VISION TRANSFORMER IMPLEMENTATION

Step 01: Import libraries

```
import torch
from torch import nn
from torch.utils.data import DataLoader
```

Step 02: Define ViT model class

```
class ViT(nn.Module):
    def __init__(self, in_dim, patch_size, num_classes):
        super(ViT, self).__init__()
        # ... (Define Vision Transformer layers here based on chosen architecture)
        self.mlp_head = nn.Sequential(
            nn.Linear(..., num_classes), # ... denotes hidden layer(s) if needed
            nn.Sigmoid() # Output probability for anomaly classification
        )
    def forward(self, x):
        # ... (Pass input through Vision Transformer layers)
        x = self.mlp_head(x)
        return x
```

Step 03: Data preprocessing (replace with specific steps for your dataset)

```
def preprocess_data(data):
    # Normalize network traffic features (e.g., scaling to 0-1 range)
    # One-hot encode categorical feature
    # ... (Apply other relevant preprocessing steps)
    return processed_data
```

Step 04: Load and preprocess training data

```
train_data = load_data("training_data.csv")
processed_train_data = preprocess_data(train_data)
```

Step 05: Define training parameters

```
learning_rate = 0.001
batch_size = 32 # Adjust based on hardware capabilities
epochs = 10
```

Step 06: Create training dataloader

```
train_loader = DataLoader(processed_train_data,
batch_size = batch_size, shuffle=True)
```

Step 07: Define loss function and optimizer (e.g., BCE-WithLogitsLoss, Adam optimizer)

```
criterion = nn.BCELoss()
optimizer = torch.optim.Adam(model.parameters(),
lr=learning_rate)
```

```
# Training loop
for epoch in range(epochs)
# ... (Training loop logic: iterate through batches, calculate loss, update weights)
```

Step 08: Evaluate model performance on validation data

Step 09: Hyperparameter Selection for Vi

The following hyperparameters were selected for ViT to optimize and improve its performance for anomaly detection:

Number of Transformer Layers: We started with a moderate number (6)–(12) and fine-tuned based on validation performance. Deeper models capture more complex relationships but can be computationally expensive.

- Hidden Dimension Size: A value in the range of 512–1024 is a common starting point. Reducing it can impact accuracy, while excessively large sizes might lead to overfitting.
- Learning Rate: A low learning rate of 0.001 was used.
- Optimizer: Adam was used for ViT training due to their efficiency in handling large models.
- Data Augmentation: Techniques like random cropping and flipping were used within normal data, aiding anomaly detection.

D. MATHEMATICAL REPRESENTATION OF THE VISION TRANSFORMER

The mathematical representation of the ViT involves several key modules: the transformer encoder, learnable embedding, and feature embedding. In the feature embedding stage, the ViT expects a sequence of token embeddings as input. After feature extraction, the features are reshaped into a flattened 2D block sequence, denoted as Y_p . Here's a modified formula that aligns with NLP concepts, where the image sequence is partitioned into multiple patches to achieve the number of patches denoted by 'p' for image classification:

$$Y_f \in \mathbb{R}^{(64) * \frac{H}{2} * \frac{W}{2}} \quad (3)$$

$$Y_p \in \mathbb{R}^{N * (P^2 C (64))} \quad (4)$$

$$N = \frac{H}{2} * \frac{W}{2} \quad (5)$$

$$Y_f \rightarrow y_p \quad (6)$$

The transformer encoder comprises two primary components: a multi-head attention block and a multi-layer perceptron (MLP) block. Normalization is performed before each block, and residual concatenation is used after each block to preserve information. The following equation demonstrates the process of obtaining multi-head attention values through

the concatenation of different head attentions:

$$head_i = (QW_i^Q, KW_i^K, VW_i^V) \quad (7)$$

$$Multihead(Q, K, V) = Concat(head_1, \dots, head_b)W^0 \quad (8)$$

Subsequently, by combining category vectors with feature block embeddings, embedding vectors are generated and used as input for the transformer encoder. Similar to convolutional neural networks, an encoder composed of blocks can extract data features beneficial for classification. The input to the transformer model is created by concatenating the category vector (X_{class}) with the embeddings for the input sequence and adding the positional encoding vector. The multi-head self-attention (MSA) and layer normalization (LN) operations are applied iteratively over the input to obtain the final output (v), which corresponds to the encoded representation of the input. This process is repeated for each layer (L) of the transformer model. The complete computational process is illustrated in the following formulas:

$$X_0 = [X_{class}; X_p^1 E], E \in \mathbb{R}^{(p^2 \cdot C) * D}, E_{pos} \in \mathbb{R}^{(N+1) * D} \quad (9)$$

$$X'_l = MSA(LN(X_l -)) + X_{l-1}, l = 1 \dots L \quad (10)$$

$$X_l = MSA(LN(X_l -)) + X'_l = 1 \dots L \quad (11)$$

$$v = LN(X_l^0) \quad (12)$$

In (9), the embedding input vector X_0 comprises both the feature embedding block $X_p^1 E$ and the category vector X_{class} . The encoding process involves multiple iterations of operations such as MAS, LN, and MLP blocks, which are repeated L times. In (10) and (11) incorporate residual connections to preserve the input information. The resulting outputs at each layer are denoted as X'_l and X_l . Finally, the feature representation obtained after the encoding process, as expressed in (12), is denoted as 'v'.

E. BAYESIAN OPTIMIZATION APPROACH TO OPTIMIZE HYPERPARAMETERS

To gain deeper insights into the design choices for our ViT-based anomaly detection model, we conducted an ablation study on key hyperparameters. This involved systematically removing or modifying individual hyperparameters while monitoring the model's performance.

We optimized the number of layers within the chosen architecture for the best balance between model complexity and performance using Bayesian optimization. The following is the Bayesian optimization approach used to optimize the hyperparameters.

Step 01: Define the search space for hyperparameters

(common examples)

```
search_space = {
```

```
'learning_rate': (0.0001, 0.1), # Common for various models
```

```
'n_estimators': (10, 100), # Relevant for ensemble methods (e.g., Random Forest)
```

```

'max_depth': (3, 50), # Decision Tree and Random
Forest
'min_samples_split': (2, 20), # Minimum samples
required to split a node
'min_samples_leaf': (1, 5), # Minimum samples
required at each leaf node
'criterion': ('gini', 'entropy'), # Splitting criterion
(information gain or Gini impurity)
'C': (0.1, 10), # SVM regularization parameter
'kernel': ('linear', 'Quadratic', 'Cubic', 'Gaussian'), #
SVM kernel type
# Hyperparameters for Neural Networks
'num_layers': (1, 10), # Number of layers (adjust for
architecture)
'neurons_per_layer': [(16, 128), (32, 256)], # List of tuples
for different layer configurations (Narrow, Medium, etc.)
'optimizer': ('adam', 'rmsprop'), # Optimizer choice
'batch_size': (32, 128), # Batch size
'dropout_rate': (0.1, 0.5), # Dropout rate
'activation': ('relu', 'tanh'), # Activation function (explore
per-layer options)
# Hyperparameters for ViT
'weight_decay': (0.0001, 0.1),
'layers': (12, 48),
'dimensions': (128, 512),
'batch_size': (20, 512)
'num_epochs': (10, 200)
# hyperparameters for LSTM
'num_layers': (1, 10), # Number of LSTM layers
'hidden_units_per_layer': (32, 512), # Hidden units per
layer
'optimizer': ('adam', 'rmsprop'), # Optimizer choice
'batch_size': (32, 256), # Batch size
'dropout_rate': (0.1, 0.5), # Dropout rate
}
Step 02: Choose a Bayesian optimization library (GPyOpt)
from bayesian_opt import BayesianOpt
Step 03: Load your training and testing data (X: features,
y: labels)
X_train, X_test, y_train, y_test = ...
Step 04: Main loop for different model types
models = ['DecisionTree', 'SVM', 'RandomForest',
'NeuralNetwork', 'LSTM', 'ViT'] # Adjust as needed
optimized_models = {}
for model_type in models:
# Update search space for specific model hyperparameters
()
Step 05: Define the function to train and evaluate the
chosen model type
def train_and_evaluate(params):
# Create the model based on the chosen type
model = get_model(model_type, params)
Step 06: Run Bayesian optimization to find the best
hyperparameters
bo = BayesianOpt(f=lambda params:
objective_function(model, X_train, X_test, y_train, y_test),

```

```

pbounds=search_space)
bo.maximize(n_iter=50)
best_params = bo.fg

```

Step 07: Train the model with the best hyperparameters
`model.fit(X_train, y_train, **best_params)`

Step 08: Train, evaluate, and store the optimized model
`optimized_models[model_type] = train_and_evaluate({})`

Step 09: Use the optimized models for further tasks
(e.g., prediction, evaluation)

IV. RESULTS AND DISCUSSIONS

The NSL-KDD dataset serves as a widely recognized benchmark dataset for network intrusion detection. It encompasses a diverse set of network traffic data, comprising various types of attacks and normal traffic. ViT represents a neural network architecture initially designed for image classification tasks but extends its applicability to various data types, including network traffic.

Data Preparation: The initial phase involves preparing the image dataset for training. This encompasses resizing the images to a uniform size, converting them into a suitable format, and partitioning them into distinct training and validation sets.

Model Architecture: Subsequently, the ViT model architecture must be precisely defined. This entails specifying the number of transformer blocks and attention heads, in addition to establishing the input image size and output classification categories.

Training: Moving forward, the model undergoes training on the prepared dataset. This training procedure involves forwarding the training data through the model, computing the loss, and applying backpropagation to adjust the model weights. Training often spans multiple epochs, with critical hyperparameters such as learning rate, batch size, and regularization being fine-tuned to maximize performance.

Validation: Following training, the model undergoes evaluation using the validation set. This step is crucial for detecting potential overfitting and gauging the model's accuracy. In cases where validation accuracy falls short of expectations, further refinements to the model architecture or hyperparameters may be necessary.

Testing: Ultimately, the model is put to the test using a distinct test set to evaluate its performance on previously unseen data. To assess the model's effectiveness, metrics such as accuracy, precision, recall, and F1 score are computed.

A. EVALUATION METRICS

In the realm of deep learning, several standard evaluation metrics come into play to gauge model performance and efficacy. Key evaluation metrics in deep learning encompass the following:

1. **Accuracy:** This metric quantifies the ratio of correctly classified samples to the total number of samples. Accuracy provides a general overview of the model's correctness in its predictions.

Precision: Precision assesses the proportion of true positives (correctly identified positive samples) relative to the sum of true positives and false positives (incorrectly identified positive samples). Precision delves into the quality of positive predictions and holds significance when the cost of false positives is high.

2. Recall (Sensitivity or True Positive Rate): Recall calculates the ratio of true positives over the sum of true positives and false negatives (positive samples mistakenly classified as negative). Emphasizing the model's ability to correctly identify positive samples, recall proves valuable when the cost of false negatives is substantial.
3. F1 Score: The F1 score emerges as the harmonic mean of precision and recall. It furnishes a balanced measure that incorporates both precision and recall, proving useful when there is an imbalance between the number of positive and negative samples in the dataset.
4. Area Under the Receiver Operating Characteristic Curve (AUC-ROC): ROC curves serve as a tool for evaluating the trade-off between true positive rate (TPR) and false positive rate (FPR) at various classification thresholds. The AUC-ROC metric quantifies the area under the ROC curve, providing a comprehensive assessment of the model's performance across diverse thresholds.
5. Mean Squared Error (MSE): MSE stands as a prevalent metric for regression tasks. It quantifies the average squared difference between predicted and actual values. Smaller MSE values signify superior model performance.

These evaluation metrics play a pivotal role in assisting researchers and practitioners in appraising the performance of deep learning models, comprehending their strengths and limitations, and comparing distinct models or algorithms. The selection of specific evaluation metrics hinges on the task at hand and the desired performance characteristics.

Table 3 shows the performance of various machine learning algorithms in classifying network traffic data from the NSL-KDD dataset. Random Forest achieved the highest overall accuracy (100%) in classifying both normal and abnormal traffic.

Adaboost (94.5%) and Naive Bayes (90.38%) follow closely. J48 appears to be an outlier with a very high but potentially unrealistic accuracy (99.78%). All algorithms performed well in identifying normal traffic, with accuracy exceeding 86.6%. Random Forest and J48 again achieved near-perfect results (99.9% and 99.8%, respectively). Random Forest and Adaboost (both 94.5%) showed the best performance in identifying abnormal traffic. J48 also performed well (99.8%), but its overall accuracy suggests potential overfitting. Naive Bayes had a lower accuracy (90.4%) for abnormal traffic detection. Random Forest and J48 have the lowest FPR (0.001 and 0.002), indicating they rarely misclassify normal traffic as abnormal. Adaboost (0.072) has a slightly higher FPR, while Naive Bayes has

the highest (0.134), meaning it's more likely to misclassify normal traffic.

Table 4 shows the performance of various machine learning algorithms on the training data from the NSL-KDD dataset. Random Forest emerges as the leader with the highest overall accuracy (99.92%). J48 follows closely (99.78%), while Adaboost (94.50%) and Naive Bayes (90.38%) lag behind. Random Forest also has the lowest error rate (0.0826% incorrectly classified). J48 has a very low error rate as well (0.2183%). Adaboost (5.49% incorrect) and Naive Bayes (9.62% incorrect) have significantly higher error rates. Kappa is a metric that considers the agreement between the model's predictions and random chance. All algorithms achieved good Kappa scores, with Random Forest (0.9983) and J48 (0.9956) having the highest values, indicating strong agreement beyond random chance. MAE (mean absolute error) and RMSE (root mean squared error) measure the average magnitude of the errors between the predicted and actual labels. Lower values indicate better performance. Here, Random Forest has the lowest values, again highlighting its accuracy. RAE (Relative Absolute Error) and RRSE (Root Relative Squared Error) are similar metrics but expressed as a percentage of the target variable's range.

Lower values are better. Random Forest has the best scores here as well.

Table 5 shows the performance of various machine learning algorithms on the test data from the NSL-KDD dataset. Random Forest again emerges as the leader with the highest overall accuracy (98.70%). J48 follows closely (98.60%), while Adaboost (90.37%) and Naive Bayes (80.73%) lag behind. All algorithms performed well in identifying normal traffic, with accuracy exceeding 69.9% (Naive Bayes). Random Forest and J48 remain at the top (both exceeding 98%). Random Forest (98.7%) and J48 (98.6%) continue to show excellent performance. Adaboost (90.4%) performs decently, but Naive Bayes struggles significantly (80.7%). Random Forest and J48 have the lowest FPR (around 0.01%), indicating a very low chance of misclassifying normal traffic as abnormal. Adaboost has a slightly higher FPR (0.107), and Naive Bayes has the highest (0.301), meaning it's most likely to misclassify normal traffic.

Table 6 shows how various machine learning algorithms performed on a 20% subset of the test data from the NSL-KDD dataset. Random Forest again takes the lead with the highest overall accuracy (97.78%). J48 follows closely (97.12%), while Adaboost (90.58%) and Naive Bayes (65.68%) fall behind significantly. All algorithms except Naive Bayes performed well in identifying normal traffic, with accuracy exceeding 61.8%. Random Forest and J48 remain at the top (both exceeding 98%). Naive Bayes continues to struggle (61.8%). Random Forest (97.8%) and J48 (97.1%) show excellent performance again. Adaboost performs decently (90.6%), but Naive Bayes lags far behind (65.7%). Random Forest has the lowest FPR (0.012), indicating a very low chance of misclassifying normal traffic. J48 has a slightly higher FPR (0.015), and Adaboost has

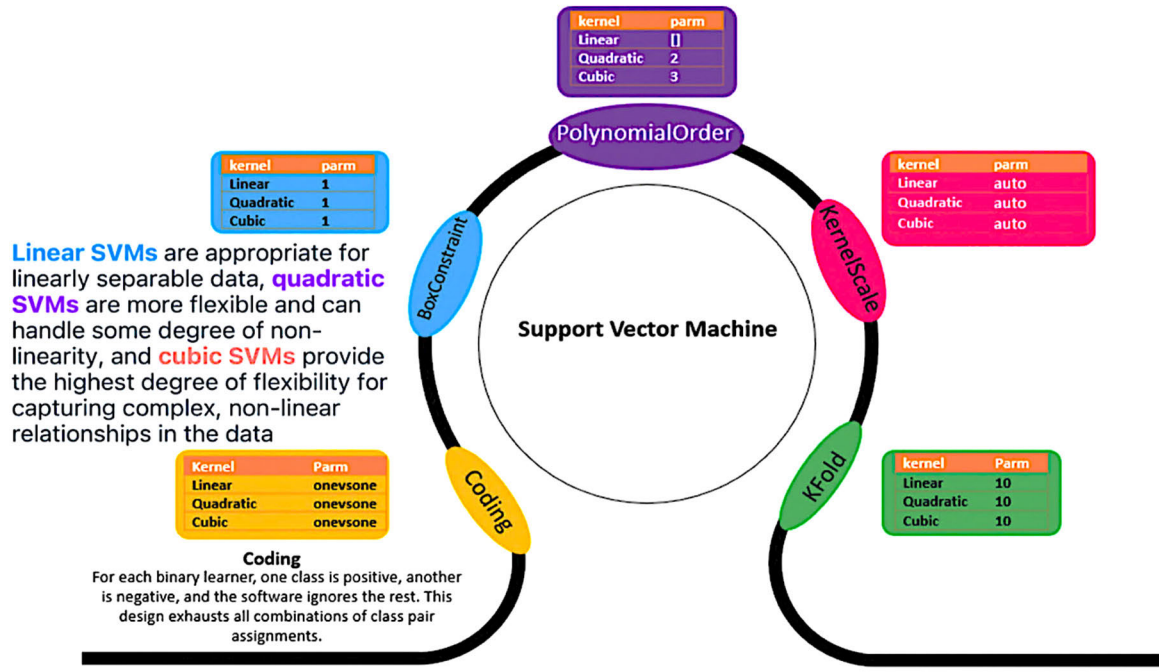


FIGURE 3. SVM Kernels.

TABLE 3. Classification result of the NSL KDD train+ percent dataset using different machine learning algorithms.

TPR	FPR	Precision	Recall	F-Measure	MCC	ROC area	Accuracy	Error	Class
Adaboost									
96%	0.072	93.90%	96%	94.90%	89%	0.988	NA	NA	Normal
92.80%	0.040	95.30%	92.80%	94%	89%	0.988	NA	NA	Abnormal
94.50%	0.057	94.50%	94.50%	94.50%	89%	0.988	94.50%	5.495	Weighted Avg.
J48									
99.80%	0.002	99.80%	99.80%	99.80%	99.60%	0.999	NA	NA	Normal
99.80%	0.002	99.80%	99.80%	99.80%	99.60%	0.999	NA	NA	Abnormal
99.80%	0.002	99.80%	99.80%	99.80%	99.60%	0.999	99.78%	0.218	Weighted Avg.
Naïve Bayes									
93.60%	0.134	89%	93.60%	91.20%	80.70%	0.967	NA	NA	Normal
86.60%	0.064	92.20%	86.60%	89.30%	80.70%	0.965	NA	NA	Abnormal
90.40%	0.101	90.50%	90.40%	90.40%	80.70%	0.966	90.38%	9.618	Weighted Avg.
Random Forest									
100%	0.001	99.90%	100%	99.90%	99.80%	1.000	NA	NA	Normal
99.90%	0.000	100%	99.90%	99.90%	99.80%	1.000	NA	NA	Abnormal
99.90%	0.001	99.90%	99.90%	99.90%	99.80%	1.000	99.91%	0.082	Weighted Avg.

NA: Not Available

a moderate FPR (0.035). Naive Bayes has the highest FPR (0.382), meaning it's most likely to misclassify normal traffic as abnormal.

Similarly, when employing J48, the weighted classification performance for detecting anomalies included a TPR of 98.60%, precision, recall, F-measure, and accuracy. Additionally, J48 achieved an MCC of 97.10% and an ROC area of 0.996.

Using Naïve Bayes, the average weighted classification performance to detect anomalies encompassed a TPR, recall,

and F-measure of 84.70%, an accuracy of 80.73%, precision of 84.40%, MCC of 65.20%, and an ROC area of 0.953.

Lastly, employing the Random Forest algorithm resulted in an average weighted classification performance for anomaly detection, including a TPR, precision, recall, F-measure, and accuracy of 98.70%, as well as an MCC of 97.40% and an ROC area of 0.974.

These findings highlight the effectiveness of the various supervised machine learning algorithms in accurately detecting anomalies within the dataset.

TABLE 4. Statistical analysis of the NSL KDD train+ dataset using different machine learning algorithms.

Measures	Adaboost	J48	Naïve Bayes	Random Forest
Correctly classified	94.50%	99.78%	90.38%	99.92%
Incorrect	5.49%	0.2183%	9.62%	0.0826%
Kappa	0.8894	0.9956	0.8059	0.9983
MAE	0.0790	0.0033	0.0965	0.0028
RMSE	0.1927	0.0457	0.3058	0.0285
RAE	15.86%	0.065%	19.40%	0.5704%
RRSE	38.62%	9.16%	61.31%	5.707%

Abbreviations: Mean Absolute error (MAE), Root mean squared error (RMSE), Relative Absolute error (RAE), Root relative squared error (RRSE)

TABLE 5. Classification result of the NSL KDD test+ percent dataset.

TPR	FPR	Precision	Recall	F-Measure	MCC	ROC area	Accuracy	Error	Class
Adaboost									
91.80%	0.107	86.70%	91.80%	89.10%	80.60%	0.968			Normal
89.30%	0.082	93.50%	89.30%	91.40%	80.60%	0.968			Abnormal
90.40%	0.093	90.50%	90.40%	90.40%	80.60%	0.968	90.37%	9.625	Weighted Avg.
J48									
98.60%	0.014	98.20%	98.60%	98.40%	97.10%	0.996			Normal
98.60%	0.014	98.90%	98.60%	98.80%	97.10%	0.996			Abnormal
98.60%	0.014	98.60%	98.60%	98.60%	97.10%	0.996	98.60%	1.401	Weighted Avg.
Naïve Bayes									
95.00%	0.301	70.50%	95%	80.90%	65.20%	0.958			Normal
69.90%	0.050	94.90%	69.90%	80.50%	65.20%	0.949			Abnormal
80.70%	0.158	84.40%	80.70%	80.70%	65.20%	0.953	80.73%	19.26	Weighted Avg.
Random Forest									
98.40%	0.011	98.50%	98.50%	98.40%	97.40%	0.974			Normal
98.90%	0.016	98.80%	98.80%	98.90%	97.40%	0.974			Abnormal
98.70%	0.014	98.70%	98.70%	98.70%	97.40%	0.974	98.70%	1.295	Weighted Avg.

TABLE 6. Classification result of the NSL KDD test -20 percent dataset.

TPR	FPR	Precision	Recall	F-Measure	MCC	ROC area	Accuracy	Error	Class
Adaboost									
63.80%	0.035	80.20%	63.80%	71.11%	66.20%	0.925			Normal
96.50%	0.362	92.30%	96.50%	94.40%	66.20%	0.925			Abnormal
90.60%	0.302	90.10%	90.60%	90.10%	66.20%	0.952	90.58%	9.426	Weighted Avg.
J48									
90.80%	0.015	93.20%	90.80%	92.00%	90.20%	0.971			Normal
98.50%	0.092	98.00%	98.50%	98.20%	90.20%	0.971			Abnormal
97.10%	0.078	97.10%	97.10%	97.10%	90.20%	0.971	97.12%	2.877	Weighted Avg.
Naïve Bayes									
83.00%	0.382	32.60%	83%	46.80%	34.70%	0.848			Normal
61.80%	0.170	94.30%	61.80%	74.70%	34.70%	0.844			Abnormal
65.70%	0.208	83.10%	65.70%	69.60%	34.70%	0.845	65.68%	34.32	Weighted Avg.
Random Forest									
93.30%	0.012	94.30%	93.30%	93.80%	92.40%	0.997			Normal
98.80%	0.067	98.50%	98.80%	98.60%	92.40%	0.997			Abnormal
97.80%	0.057	97.80%	97.80%	97.80%	92.40%	0.997	97.78%	2.236	Weighted Avg.

Fig. 3 reflects the SVM with different kernel functions along with other parameters, which were optimized using

the Bayesian optimization algorithm. These findings highlight the effectiveness of the various supervised machine

learning algorithms in accurately detecting anomalies within the dataset.

Table 7 compares the performance of various machine learning algorithms on the NSL KDD Train+ percent dataset using Bayesian optimization for parameter tuning.

The Fine Tree excels with exceptional performance across metrics by yielding sensitivity (attack detection) very high (likely exceeding 99%), specificity (avoiding false positives) also very high (likely exceeding 99%), accuracy (overall correctness) almost perfect (around 99.81%), AUC (classification strength) very strong (around 0.9989), F1-Score (balanced accuracy) very high (around 99.64%), and MCC (balanced classification quality) very high (around 99.62%). The medium and coarse trees offer a good accuracy-efficiency balance by maintaining reasonably high accuracy (around 98–99%) and F1-Score (around 96–98%). Potentially require less training time, making them efficient for resource-constrained scenarios. The SVM linear offers a good balance, i.e., high accuracy (around 97.66%) and efficiency (potentially faster training than complex models). Strong overall classification ability with a high AUC (around 0.9955). The SVM quadratic achieves the highest AUC (around 0.9993), suggesting excellent classification. SVM Medium Gaussian demonstrates another strong performer with high accuracy (around 99.28%) and F1-Score (around 98.33%), potentially offering a good balance between performance and complexity. Boosted Tree and Bagged Tree achieve exceptional performance, exceeding even Fine Tree in some aspects. Accuracy reached near-perfect levels (around 99.7–99.9%) and high F1-Scores (around 99.5–99.8%). This highlights the effectiveness of ensemble methods in combining multiple weaker models for improved robustness and accuracy. RUSBoosted Tree exhibits good performance while potentially being faster to train. Accuracy remains high (around 98.78%). It utilizes random under-sampling (reducing training data size) for potentially faster training times. All neural network architectures (narrow, medium, bilayered, and trilayered) demonstrate consistently high performance, i.e., accuracy exceeding 99.6% across all architectures. Strong AUC and F1-Score values. This showcases their capability for complex pattern recognition in this dataset. The LSTM achieves near-perfect performance, with all metrics at or very close to 100%. This suggests an exceptional ability to identify normal and attack instances in the dataset. The optimized bidirectional ViT achieves perfect performance with all metrics at 100%. This indicates outstanding capability in classifying normal and attack instances, potentially even surpassing LSTM in this specific task.

Table 7 presents the performance of a Vision Transformer model used for anomaly detection. The model appears to have two main parts: feedforward_0 and feedforward_1. These are likely the core Vision Transformer layers that process the input data. MLP followed by a sigmoid layer: This final part is likely a Multi-Layer Perceptron (MLP) used to classify the input as normal or anomalous based on the features extracted

by the Vision Transformer. The sigmoid layer outputs a probability between 0 and 1, with values closer to 1 indicating a higher likelihood of an anomaly. The model achieved perfect accuracy (100%) on the training data. This is often a red flag, as it suggests the model might be overfitting to the training data and may not perform well on unseen data. The validation accuracy, which reflects performance on a separate dataset not used for training, is significantly lower (78.80%). The model has a relatively small number of parameters (around 46,000).

This could be a positive aspect if the goal is to have a lightweight and efficient model. However, a small number of parameters can also limit the model's ability to learn complex patterns, potentially affecting its accuracy.

Table 8 focuses on the computational complexity of various algorithms used to classify the NSL KDD Train+ dataset with Bayesian parameter optimization. Tree-based classifiers (fine, medium, and coarse trees) offer a good balance, i.e., training time is relatively fast (ranging from 66.10 to 76.73 seconds). The model size is medium (ranging from 21 kB to 53 kB). The prediction speed is fast (ranging from 55,000 to 61,000 observations per second). The accuracy (validation) yielded was high, with Fine Tree achieving the highest (99.8%). As tree complexity increases (fine, medium, or coarse), training time increases slightly, while accuracy decreases modestly. Support Vector Machines (SVMs) show a wider range of complexity. SVM Linear has the fastest training time (3561 seconds) and smallest model size (7 MB), but lower accuracy (97.7%) compared to some other models. SVM quadratic yielded Higher accuracy (99.5%) but significantly longer training time (6931 seconds) and smaller model size (2 MB) compared to some neural networks. SVM Cubic provided poor performance (47.4% accuracy) despite moderate training time and model size.

Not suitable for this task. SVM Medium Gaussian offers a balance with good accuracy (99.3%), moderate training time (2008 seconds), and model size (4 MB), but slower prediction speed (3100 obs/sec). Boosted Tree, Bagged Tree, and RUSBoosted Tree maintain a good balance, i.e., training time is relatively fast (ranging from 230 to 351 seconds), model size is moderate (ranging from 559 kB to 2 MB), prediction speed is decent (ranging from 19,000 to 21,000 obs/sec), and accuracy is high (ranging from 98.8% to 99.9%). Narrow, Medium, Bilayered, and Trilayered NN offer high accuracy (all above 99.6%) but require more resources. The training time is significantly longer (ranging from 7282 to 9162 seconds). The model size remains small (ranging from 32 kB to 47 kB). Prediction speed is slower compared to simpler models (ranging from 7300 to 14,000 obs/sec). There are minimal performance differences between the architectures, suggesting further analysis might be needed for optimal efficiency selection. Deep learning models, i.e., bidirectional LSTM, achieve high accuracy (99.8%) with moderate training time (4560 seconds) and model size (590 MB). Prediction speed is decent (14,000 obs/sec). Vision Transformer achieves perfect accuracy (100%) with the longest training time

TABLE 7. Classification result of the NSL KDD train+ percent dataset using different ML, neural networks, and deep learning algorithms with bayesian optimization parameters and an optimization approach.

Method	Sensitivity	Specificity	PPV	NPV	Accuracy	AUC	F1-Score	MCC
01) Tree based classifiers								
Fine Tree	99.84%	99.79%	99.76%	99.86%	99.81%	0.9989	99.64%	99.62%
Medium Tree	98.05%	99.43%	99.35%	98.28%	98.78%	0.9946	96.77%	97.55%
Coarse Tree	96.82%	96.58%	96.11%	97.21%	96.69%	0.9676	93.40%	93.36%
02) Support Vector Machine (SVM) classifiers								
SVM Linear	97.86%	97.48%	97.09%	98.15%	97.66%	0.9955	95.38%	95.29%
SVM Quadratic	99.45%	99.51%	99.44%	99.52%	99.48%	0.9993	98.90%	98.96%
SVM Cubic	46.76%	56.92%	94.585	6.24%	47.35%	0.4898	29.26%	1.73%
Medium Gaussian	99.10%	99.44%	99.36%	99.21%	99.28%	0.9991	98.33%	98.56%
03) Ensemble methods								
Boosted Tree	99.86%	99.62%	99.57%	99.88%	99.73%	0.9998	99.57%	99.46%
Bagged Tree	99.90%	99.89%	99.87%	99.91%	99.90%	1.00	99.78%	99.78%
RUSBoosted Tree	98.07%	99.41%	99.33%	98.30%	98.78%	0.9969	96.80%	97.56%
04) Neural Networks (NN)								
Narrow NN	99.54%	99.64%	99.59%	99.60%	99.60%	0.9994	99.11%	99.19%
Medium NN	99.70%	99.70%	99.66%	99.74%	99.70%	0.9995	99.38%	99.40%
Bilayered NN	99.67%	99.65%	99.59%	99.71%	99.66%	0.9993	99.30%	99.31%
Trilayered NN	99.69%	99.62%	99.56%	99.73%	99.65%	0.9994	99.31%	99.30%
05) Optimized LSTM								
LSTM	99.98%	99.91%	99.98%	99.91%	99.97%	1.00	99.96%	99.89%
06) Optimized Bidirectional ViT								
ViT	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

TABLE 8. Computational complexity to classify the NSL KDD train+ dataset using different ML, neural networks, and deep learning algorithms with bayesian optimization parameters and an optimization approach.

Model	Training time	Model Size	Prediction Speed	Accuracy (Validation)	Total cost (Validation)
Fine Tree	76.73 sec	53kB	60000 obs/sec	99.8%	238
Medium Tree	73.32 sec	26 kB	61000 obs/sec	98.8%	1540
Coarse Tree	66.10 sec	21 kB	55000 obs/sec	96.7%	4133
SVM Linear	3561 sec	7 MB	8400 obs/sec	97.7%	2954
SVM Quadratic	6931 sec	2 MB	47000 obs/sec	99.5%	650
SVM Cubic	8434 sec	160 kB	80000 obs/sec	47.4%	66322
SVM Medium Gaussian	2008 sec	4 MB	3100 obs/sec	99.3%	906
Boosted Tree	230 sec	804 kB	19000 obs/sec	99.7%	337
Bagged Tree	351 sec	2 MB	21000 obs/sec	99.9%	135
RUSBoosted Tree	272 sec	559 kB	19000 obs/sec	98.8%	1537
Narrow NN	7282 sec	32 kB	7800 obs/sec	99.6%	510
Medium NN	9162 sec	47 kB	14000 obs/sec	99.7%	377
Bilayered NN	7966 sec	34 kB	7300 obs/sec	99.7%	431
Trilayered NN	8452 sec	36 kB	13000 obs/sec	99.7%	440
Bidirectional LSTM	4560 sec	590 MB	14000 obs/sec	99.8%	982
Vision Transformer	10490 sec	790 MB	25000 obs/sec	100%	15400

(10490 seconds) and largest model size (790 MB). Prediction speed is moderate (25,000 obs/sec).

Fig. 4 displays the training and validation accuracy loss graph generated using vision transfer for anomaly detection across 500 epochs. Notably, the training accuracy stabilizes at 100% after a few initial epochs, while the validation accuracy reaches 78.43% after the entire 500 epochs. In the first epoch,

the model achieved an accuracy of 94.35%, with a validation loss of 0.8961 and a validation accuracy of 77.28%. Subsequently, after 500 epochs, the model maintained a training accuracy of 100% and a validation accuracy of 78.43%.

Fig. 5 shows the AUC of selected classifiers to distinguish normal subjects from intrusion-attack subjects. The AUC was obtained as follows: Fig. 5 a) fine tree (AUC = 0.9989), Fig. 5

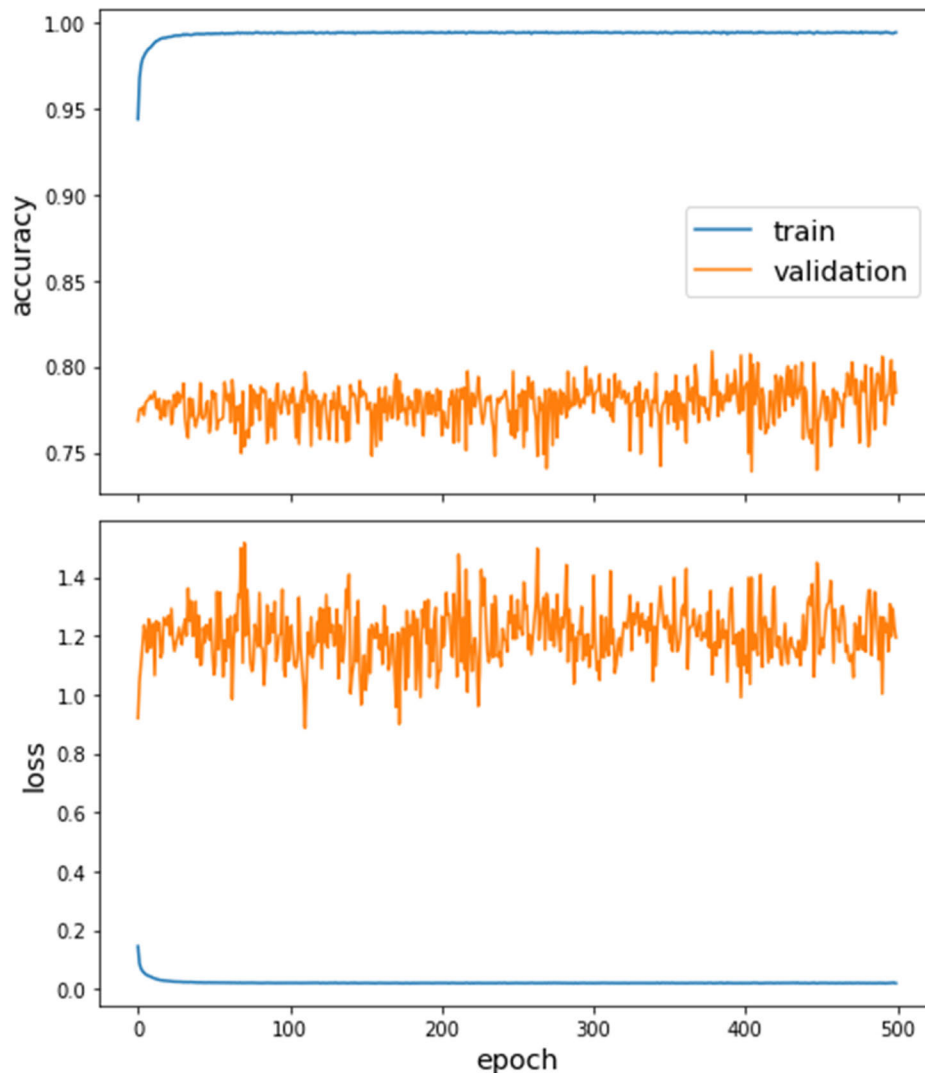


FIGURE 4. Accuracy-loss graph using ViT for anomaly detection.

TABLE 9. Classification accuracy using the CNN sequential network for anomaly detection.

Layer	Type	Output shape	Param #	Training accuracy	Validation accuracy
lstm	(LSTM)	(None, 80)	26,240	100%	56.92%
dense	(Dense)	(None, 1)	81		
Total params:			26,321		
Trainable params:			26,321		

b) SVM linear (AUC = 0.9955), Figure 5 c) SVM cubic (AUC = 0.4898), Figure 5 d) RUSBoosted (AUC = 0.9969), and Figure 5 e) trilayered NN (AUC = 0.9994).

Fig. 6 reflects the classification performance using a selected fine tree classifier to detect intrusions in the NSL-KDD train+ dataset. Figure 6a shows the confusion matrix. The positive (Anomaly) class contains 58630 subjects, while the negative (normal) class contains 67343 for a total of

125,973 subjects. After applying the fine tree, there were TP = 58487, FP = 143, FN = 95, and TN = 67248. Figure 6b) indicates a TPR of 99.8% and a FPR of 99.9%. While Figure 6 c) reflects the PPV and NPV of 99.8%, respectively.

Table 9 provides an overview of the training and validation accuracy, as well as details about the architecture, layers, types, and number of parameters utilized in the CNN

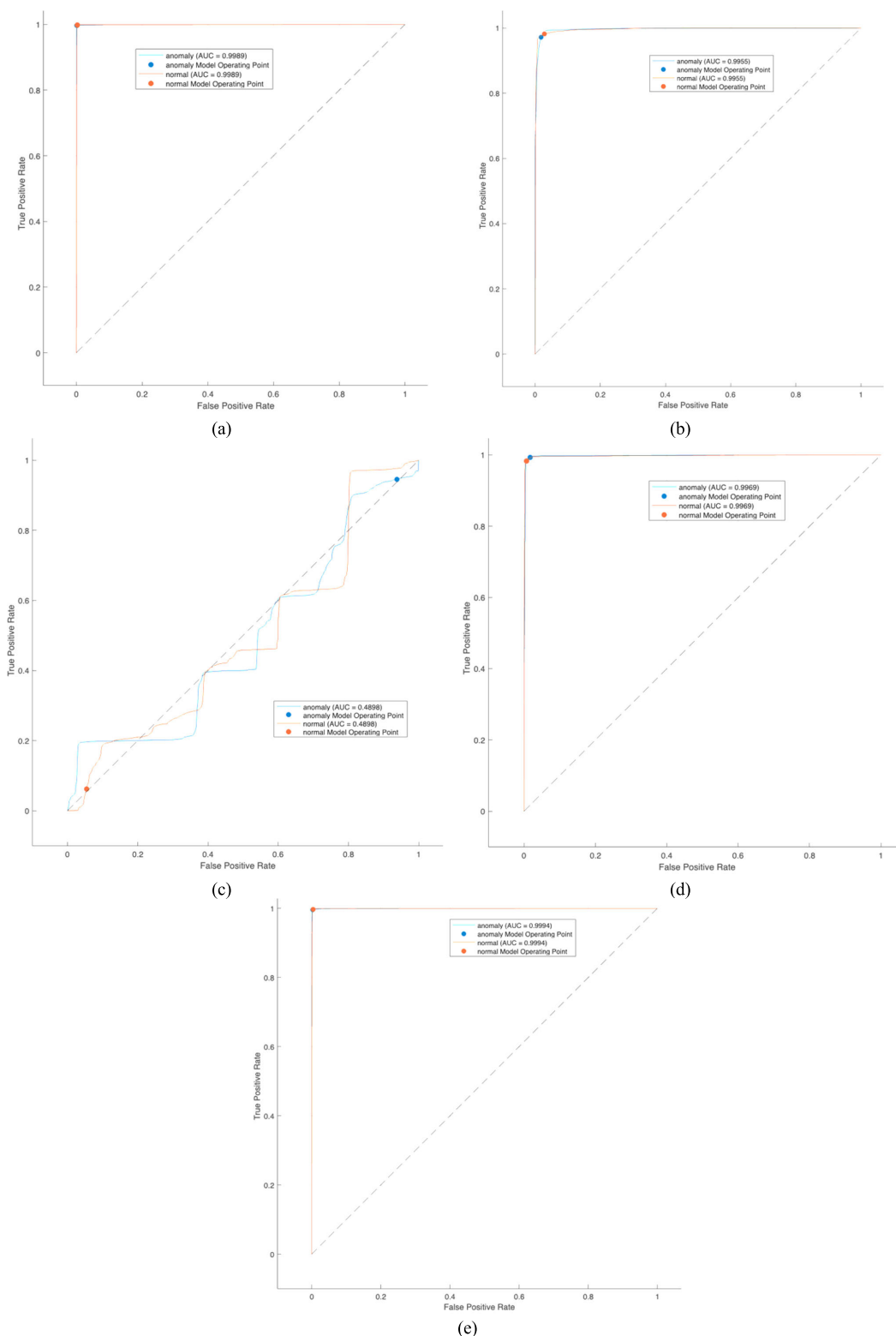


FIGURE 5. AUC to detect anomalies by utilizing ML and NNs a) fine tree, b) SVM linear, c) SVM cubic, d) RUSBoosted, e) Trilayered NN.

sequential model. The model employed a total of 26,321 parameters, achieving a validation accuracy of 56.92% and a training accuracy of 100%.

In Table 10, the classification accuracy results are presented for the ViT. This model utilized a total of 46,925 parameters, out of which 46,753 were trainable. The ViT

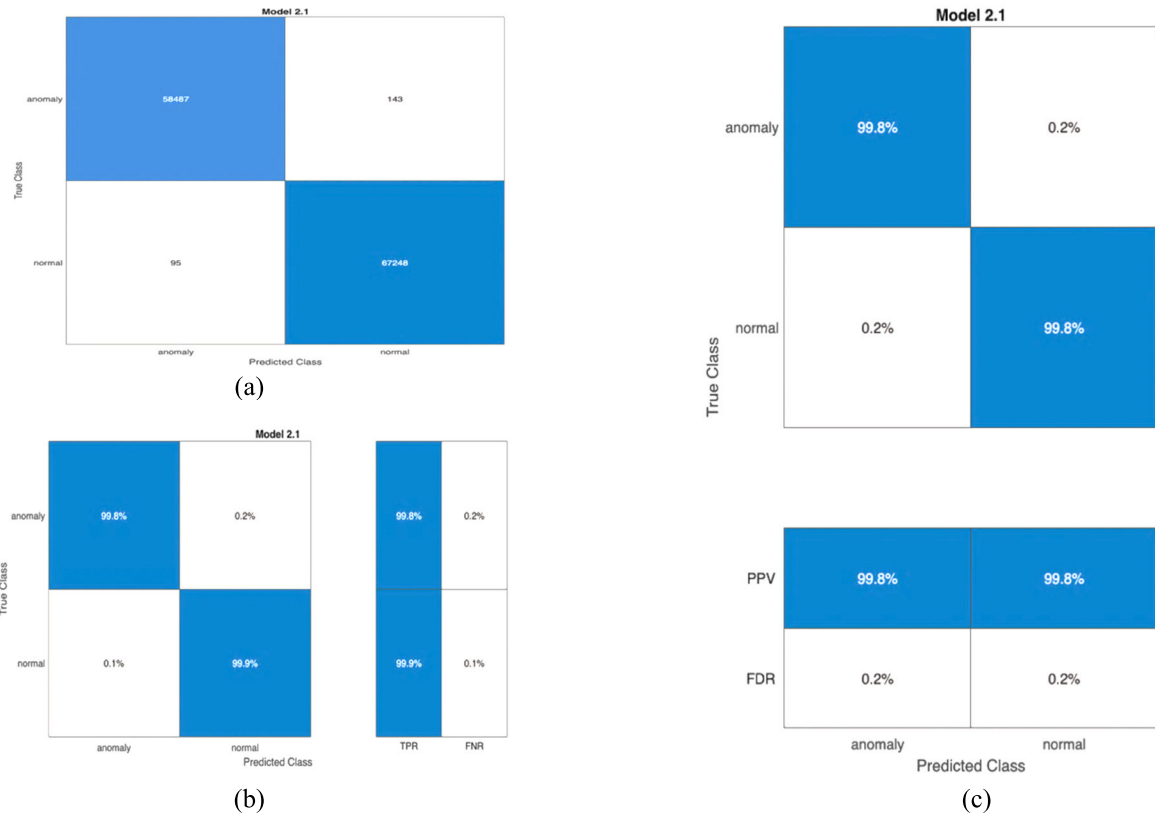


FIGURE 6. Classification performance using a machine learning fine tree: a) confusion matrix; b) TPR-FPR; c) PPV and NPV.

achieved a training accuracy of 100% and a validation accuracy of 78.80%.

These tables offer insights into the performance and complexity of both the CNN sequential model and the ViT model, showcasing their respective training and validation accuracies and the number of parameters employed.

Based on these comprehensive findings, it is evident that the proposed method holds promise for implementation in IoT networks, offering an effective means of detecting a wide range of network attacks.

V. DISCUSSION

The NSL-KDD dataset stands as an improved iteration of the widely utilized KDD Cup 1999 dataset, specifically tailored for intrusion detection systems. It addresses various limitations observed in the KDD Cup 1999 dataset, including the presence of redundant and irrelevant features as well as the use of pre-selected training and testing sets. The NSL-KDD dataset has emerged as a pivotal benchmark dataset for evaluating intrusion detection systems' performance. It encompasses network traffic data encompassing a range of attack types, including DoS, Probe, R2L, and U2R attacks. Moreover, the dataset encompasses both binary and multi-class classification challenges, rendering it suitable for a wide array of intrusion detection approaches.

Leveraging the NSL-KDD dataset empowers researchers and practitioners to conduct standardized comparisons of

various intrusion detection algorithms and techniques. This standardization aids in the development of more accurate and effective intrusion detection systems, which are integral to safeguarding computer networks against cyberattacks. The dataset encapsulates the intricate and concealed dynamics inherent to intrusion detection systems, driving researchers to devise tools for enhancing predictions.

In this study, we embarked on a journey to harness and optimize machine-learning algorithms. Subsequently, we delved into the realm of deep learning, exploring the potential of Vision Transformer algorithms. Our findings indicate that the proposed approach, especially the vision transformer, exhibits robustness in anomaly detection. This robustness can be harnessed for advanced decision-making, early detection, automated response mechanisms, compliance adherence, and safeguarding against emerging threats. IDS stands poised to play an indispensable role in the protection of computer networks and systems against unauthorized access and malicious attacks. These systems meticulously scrutinize IoT network traffic and system behavior, promptly notifying administrators or initiating automated countermeasures to thwart or mitigate attacks.

Looking forward, future research endeavors should prioritize model optimization, real-time implementation, and enhancing adversarial robustness within intrusion detection systems. Addressing these research facets will propel the field of intrusion anomaly detection employing vision

TABLE 10. Classification accuracy using the vision transformer for anomaly detection.

Layer Type Connected to	Output shape	Param #	Training accuracy	Validation accuracy
feedforward_0 (Sesquential) ['concatenate [0][0]']	(None, 86)	7,654	100.00%	78.80%
feedforward_1 (Sequential) ['feedforward_0[0][0]']	(None, 86)	7,654		
MLP (Sequential) ['feedforward_1[0][0]']	(None, 86)	30,186		
sigmoid (Dense) ['MLP[0][0]']	(None, 1)	87		
Total params:		46,925		
Trainable params:		46,753		

transformers within the IoT cyber landscape. Such advancements will significantly contribute to bolstering the security and trustworthiness of IoT networks.

ViT is emerging as a powerful tool for anomaly detection in traffic sequences, potentially surpassing CNNs and RNNs due to their unique properties. ViT's secret weapon lies in self-attention. Unlike CNNs with limited receptive fields, ViT can analyze the entire traffic sequence at once, capturing long-range interactions between distant elements. This is crucial for spotting anomalies that might involve vehicles spread across multiple lanes. While not inherently sequential like RNNs, ViT can be adapted to understand the order of information within the traffic flow using positional encoding. This allows the model to learn the temporal relationships between traffic patterns, further aiding in anomaly detection. Furthermore, ViT's flexibility shines when dealing with traffic data that goes beyond fixed grids, such as sensor readings or GPS coordinates. It can directly process these sequences, making it adaptable to various data formats collected by traffic monitoring systems. Finally, pre-trained ViT models, honed on massive image datasets, can be fine-tuned for traffic anomaly detection tasks. This transfer learning leverages prior knowledge to improve performance, especially when dealing with limited anomaly-specific datasets. The ViT's capability to model long-range dependencies, along with its data format flexibility, makes it a promising tool for traffic anomaly detection. While CNNs and RNNs are established players, ViT's potential for exploiting global context and transfer learning positions it as a compelling avenue for further exploration in this field.

Based on all these metrics, Random Forest emerges as the best-performing algorithm for this dataset. It achieves the highest accuracy, the lowest error rates, and the strongest agreement beyond random chance. J48 follows closely, with excellent performance as well. Adaboost performs decently, while Naive Bayes struggles with a higher error rate.

Fine Tree achieves excellent overall performance with high sensitivity, specificity, accuracy, AUC, F1-Score, and MCC. These metrics indicate a well-balanced classifier that effectively identifies both normal and attack instances. Medium- and coarse-scale trees offer a trade-off between accuracy and computational efficiency. They perform reasonably well with high accuracy and F1-Score, potentially making them faster to train and use. SVM linear offers a good balance between accuracy and computational efficiency.

Its AUC is high, indicating good overall performance in distinguishing between normal and attack instances. SVM quadratic achieves the highest AUC, suggesting excellent classification ability. However, it might come at the cost of increased training time and complexity. SVM Medium Gaussian demonstrates another strong performer with high accuracy, F1-Score, and AUC, possibly offering a good balance between performance and complexity.

Boosted Tree and Bagged Tree achieve exceptional performance, exceeding even Fine Tree in some respects. This highlights the effectiveness of ensemble methods in combining multiple weaker models for improved robustness and accuracy. RUSBoosted Tree exhibits good performance while potentially being faster to train than Boosted Tree due to the use of random undersampling. All neural network architectures (narrow, medium, bilayered, and trilayered) demonstrate remarkable performance with high accuracy, AUC, and F1-Score. This showcases the capability of neural networks for complex pattern recognition in this dataset. LSTM achieves near-perfect performance, with all metrics at or very close to 1.0. This suggests an exceptional ability to identify normal and attack instances in the dataset. ViT achieves perfect performance, with all metrics at 1.0. This indicates outstanding capability in classifying normal and attack instances, potentially even surpassing LSTM in this specific task. Ensemble methods (Boosted Tree and Bagged Tree) and optimized deep learning models (LSTM and ViT) achieve the highest overall performance based on the combination of metrics. SVM and neural networks also exhibit strong performance, indicating their suitability for this type of classification task. Tree-based classifiers offer a good balance of accuracy and efficiency for less resource-intensive applications. The choice of algorithm depends on the trade-off between accuracy, computational cost, interpretability, and other factors relevant to your specific needs.

The choice of algorithm depends on the trade-off between accuracy, computational resources (training time, model size), and prediction speed. Tree-based classifiers and some SVMs offer a good balance for applications where speed and efficiency are crucial. Ensemble methods provide high accuracy while maintaining reasonable resource requirements. Neural networks achieve excellent accuracy, but they require more computational resources. Deep learning models (e.g., LSTM and ViT) offer the highest accuracy but come with the highest resource demands. Consider them for tasks

where maximizing accuracy is paramount and computational resources are abundant.

VI. CONCLUSION

The Internet of Things (IoT) landscape presents a double-edged sword. While increased connectivity fosters innovation, it also expands the attack surface for cybercriminals. Intrusions on IoT devices can have devastating consequences, disrupting operations and compromising sensitive data. Robust intrusion detection systems (IDS) are essential to address this growing threat.

However, data generated by IoT devices poses a significant challenge for anomaly detection. This data is often multivariate (containing multiple variables), highly complex, non-stationary (its statistical properties change over time), and non-linear. Traditional methods struggle to extract meaningful patterns from such intricate data.

This research explores a novel approach to intrusion detection in IoT networks. We leverage and optimize machine learning, neural networks, and deep learning techniques to enhance anomaly detection. Furthermore, we employed Bayesian optimization to optimize the hyperparameters of these algorithms, aiming to further improve prediction performance. For evaluation, we utilized the NSL-KDD dataset, which comprises training data, testing data with a 20% positive class imbalance, and testing data with a 20% negative class imbalance, both containing multivariate intrusion and normal data points.

Our machine learning models, employing both tree-based and ensemble approaches, achieved impressive performance. Notably, the medium and coarse tree-based models, with training times of 66.10 seconds and memory footprints of 21 kB, offered a good balance between performance and computational efficiency. This makes them suitable for implementation on resource-constrained edge devices. While optimized neural networks and deep learning models like LSTMs and ViTs achieved perfect detection rates, they also demanded significantly more computational resources. These findings highlight the potential of our proposed methods for enhancing attack and intrusion detection in resource-limited IoT networks. Improved detection capabilities are crucial for safeguarding system integrity, identifying fraudulent activity, and optimizing overall system performance. Our approach presents a promising solution for securing the ever-evolving landscape of IoT networks.

VII. LIMITATION AND FUTURE DIRECTION

ViT models can be computationally expensive compared to CNNs, especially for real-time traffic monitoring applications. Additionally, while ViT can be adapted for sequential data, it might not capture temporal dependencies as effectively as specialized RNN architectures. Future work should explore techniques for improving ViT's computational efficiency while maintaining its strengths in anomaly detection. Research into incorporating RNN-inspired architectures within the ViT framework to enhance its ability to

learn complex temporal relationships in traffic sequences is also promising. By addressing these limitations, ViT has the potential to become a dominant force in traffic anomaly detection. While ViTs offer promising results for anomaly detection, they also have limitations. Training ViTs can be computationally expensive compared to simpler models like CNNs. This can be a bottleneck for real-time anomaly detection on resource-constrained devices. We can explore techniques for reducing the computational cost of ViTs, such as model pruning or quantization, to enable deployment on edge devices. By addressing these limitations and exploring future research directions, we can further improve the efficacy and practicality of ViT-based anomaly detection for network traffic data.

We conducted a comprehensive analysis of the intrusion detection dataset, employing various machine learning and deep learning techniques with hyperparameter optimization through Bayesian optimization, and compared the results with those of recent studies. This approach allowed for a deeper analysis of different NSL-KDD datasets, including NSL-KDD train+, NSL-KDD test+20%, and NSL-KDD test-20%. All datasets comprise multivariate information for binary class classification. Our future work will involve applying and validating these methods on broader anomaly detection datasets with multi-class classification.

REFERENCES

- [1] S. M. Taghavinejad, M. Taghavinejad, L. Shahmiri, M. Zavvar, and M. H. Zavvar, "Intrusion detection in IoT-based smart grid using hybrid decision tree," in *Proc. 6th Int. Conf. Web Res. (ICWR)*, Apr. 2020, pp. 152–156, doi: [10.1109/ICWR49608.2020.9122320](https://doi.org/10.1109/ICWR49608.2020.9122320).
- [2] R. U. Khan, J. Yin, F. S. Mustafa, and S. Wang, "Analyzing human factor involvement in sustainable hazardous cargo port operations," *Ocean Eng.*, vol. 250, Apr. 2022, Art. no. 111028, doi: [10.1016/j.oceaneng.2022.111028](https://doi.org/10.1016/j.oceaneng.2022.111028).
- [3] P. De Giovanni, V. Belvedere, and A. Grando, "The selection of industry 4.0 technologies through Bayesian networks: An operational perspective," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 2921–2936, 2024, doi: [10.1109/TEM.2022.3200868](https://doi.org/10.1109/TEM.2022.3200868).
- [4] K. Amzil, E. Yahia, N. Klement, and L. Roucoules, "Automatic neural networks construction and causality ranking for faster and more consistent decision making," *Int. J. Comput. Integr. Manuf.*, vol. 36, no. 5, pp. 735–755, May 2023, doi: [10.1080/0951192x.2022.2134930](https://doi.org/10.1080/0951192x.2022.2134930).
- [5] V. P. Gandhi, N. S. L. Jatla, G. Sadhineni, S. Geddamura, G. K. Chaitanya, and A. K. Velmurugan, "A comparative study of AI algorithms for anomaly-based intrusion detection," in *Proc. 7th Int. Conf. Comput. Methodologies Commun. (ICCMC)*, Feb. 2023, pp. 530–534, doi: [10.1109/ICCMC56507.2023.10084186](https://doi.org/10.1109/ICCMC56507.2023.10084186).
- [6] A. Ali, A. W. Septyanto, I. Chaudhary, H. A. Hamadi, H. M. Alzoubi, and Z. F. Khan, "Applied artificial intelligence as event horizon of cyber security," in *Proc. Int. Conf. Bus. Analytics for Technol. Secur. (ICBATS)*, Feb. 2022, pp. 1–7, doi: [10.1109/ICBATS54253.2022.9759076](https://doi.org/10.1109/ICBATS54253.2022.9759076).
- [7] W. Jiang, "Cellular traffic prediction with machine learning: A survey," *Exp. Syst. Appl.*, vol. 201, Sep. 2022, Art. no. 117163, doi: [10.1016/j.eswa.2022.117163](https://doi.org/10.1016/j.eswa.2022.117163).
- [8] A. S. Ibrahim, A. M. Abbas, A. M. A. Hassan, W. M. F. Abdel-Rehim, A. Emam, and S. Mohsen, "Design and implementation of a pilot model for IoT smart home networks," *IEEE Access*, vol. 11, pp. 59701–59728, 2023, doi: [10.1109/ACCESS.2023.3282095](https://doi.org/10.1109/ACCESS.2023.3282095).
- [9] S. Tiwari, A. Bhushan, A. K. Singh, and R. K. Yadav, "Unleashing the potential of IoT integration for energy optimization in smart homes," in *Proc. 3rd Int. Conf. Power Electron. IoT Appl. Renew. Energy Control (PARC)*, Feb. 2024, pp. 82–85, doi: [10.1109/parc59193.2024.10486624](https://doi.org/10.1109/parc59193.2024.10486624).

- [10] E. Korneeva, N. Oliner, and W. Strielkowski, "Consumer attitudes to the smart home technologies and the Internet of Things (IoT)," *Energies*, vol. 14, no. 23, p. 7913, Nov. 2021, doi: [10.3390/en14237913](https://doi.org/10.3390/en14237913).
- [11] B. Radenkovic and P. Kocovic, "From ubiquitous computing to the Internet of Things," in *Securing the Internet of Things*. Hershey, PA, USA: IGI Global, 2019, pp. 1523–1556.
- [12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015, doi: [10.1109/COMST.2015.2444095](https://doi.org/10.1109/COMST.2015.2444095).
- [13] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of threats to the Internet of Things," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1636–1675, 2nd Quart., 2019, doi: [10.1109/COMST.2018.2874978](https://doi.org/10.1109/COMST.2018.2874978).
- [14] I. Stelliou, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of IoT-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3453–3495, 4th Quart., 2018, doi: [10.1109/COMST.2018.2855563](https://doi.org/10.1109/COMST.2018.2855563).
- [15] B. B. Zarpelão, 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, doi: [10.1016/j.jnca.2017.02.009](https://doi.org/10.1016/j.jnca.2017.02.009).
- [16] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 2046–2069, 4th Quart., 2013, doi: [10.1109/SURV.2013.031413.00127](https://doi.org/10.1109/SURV.2013.031413.00127).
- [17] N. Moustafa, N. Koroniotis, M. Keshk, A. Y. Zomaya, and Z. Tari, "Explainable intrusion detection for cyber defences in the Internet of Things: Opportunities and solutions," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 3, pp. 1775–1807, 3rd Quart., 2023, doi: [10.1109/COMST.2023.3280465](https://doi.org/10.1109/COMST.2023.3280465).
- [18] A. Heidari and M. A. Jabraeil Jamali, "Internet of Things intrusion detection systems: A comprehensive review and future directions," *Cluster Comput.*, vol. 26, no. 6, pp. 3753–3780, Dec. 2023, doi: [10.1007/s10586-022-03776-z](https://doi.org/10.1007/s10586-022-03776-z).
- [19] S. A. Bakhsh, M. A. Khan, F. Ahmed, M. S. Alshehri, H. Ali, and J. Ahmad, "Enhancing IoT network security through deep learning-powered intrusion detection system," *Internet Things*, vol. 24, Dec. 2023, Art. no. 100936, doi: [10.1016/j.iot.2023.100936](https://doi.org/10.1016/j.iot.2023.100936).
- [20] S. V. N. S. Kumar, M. Selvi, and A. Kannan, "A comprehensive survey on machine learning-based intrusion detection systems for secure communication in Internet of Things," *Comput. Intell. Neurosci.*, vol. 2023, pp. 1–24, Jan. 2023, doi: [10.1155/2023/8981988](https://doi.org/10.1155/2023/8981988).
- [21] N. J. Singh, N. Hoque, K. R. Singh, and D. K. Bhattacharyya, "Botnet-based IoT network traffic analysis using deep learning," *Secur. Privacy*, vol. 7, no. 2, p. 355, Mar. 2024, doi: [10.1002/spy2.355](https://doi.org/10.1002/spy2.355).
- [22] B. I. Mukhtar, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, "IoT vulnerabilities and attacks: SILEX malware case study," *Symmetry*, vol. 15, no. 11, p. 1978, Oct. 2023, doi: [10.3390/sym15111978](https://doi.org/10.3390/sym15111978).
- [23] B. Sutheshkan, S. Basheer, G. Thangavel, and O. P. Sharma, "Evolution of malware targeting IoT devices and botnet formation," in *Proc. IEEE Int. Conf. Comput., Power Commun. Technol. (IC2PCT)*, Feb. 2024, pp. 1415–1422, doi: [10.1109/ic2pct60090.2024.10486705](https://doi.org/10.1109/ic2pct60090.2024.10486705).
- [24] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use AI to enhance security?" *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 41–49, Sep. 2018, doi: [10.1109/MSP.2018.2825478](https://doi.org/10.1109/MSP.2018.2825478).
- [25] E. Anthei, L. Williams, M. Slowinska, G. Theodorakopoulos, and P. Burnap, "A supervised intrusion detection system for smart home IoT devices," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 9042–9053, Oct. 2019, doi: [10.1109/JIOT.2019.2926365](https://doi.org/10.1109/JIOT.2019.2926365).
- [26] W. Li, W. Meng, and M. H. Au, "Enhancing collaborative intrusion detection via disagreement-based semi-supervised learning in IoT environments," *J. Netw. Comput. Appl.*, vol. 161, Jul. 2020, Art. no. 102631, doi: [10.1016/j.jnca.2020.102631](https://doi.org/10.1016/j.jnca.2020.102631).
- [27] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai, and C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Exp. Syst. Appl.*, vol. 38, no. 1, pp. 306–313, Jan. 2011, doi: [10.1016/j.eswa.2010.06.066](https://doi.org/10.1016/j.eswa.2010.06.066).
- [28] A. S. Eesa, Z. Orman, and A. M. A. Brifcani, "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," *Exp. Syst. Appl.*, vol. 42, no. 5, pp. 2670–2679, Apr. 2015, doi: [10.1016/j.eswa.2014.11.009](https://doi.org/10.1016/j.eswa.2014.11.009).
- [29] R. M. A. Mohammad and M. K. Alsmadi, "Intrusion detection using highest wins feature selection algorithm," *Neural Comput. Appl.*, vol. 33, no. 16, pp. 9805–9816, Aug. 2021, doi: [10.1007/s00521-021-05745-w](https://doi.org/10.1007/s00521-021-05745-w).
- [30] J. Li, Z. Zhao, R. Li, and H. Zhang, "AI-based two-stage intrusion detection for software defined IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2093–2102, Apr. 2019, doi: [10.1109/JIOT.2018.2883344](https://doi.org/10.1109/JIOT.2018.2883344).
- [31] Z. Tian, M. Zhuo, L. Liu, J. Chen, and S. Zhou, "Anomaly detection using spatial and temporal information in multivariate time series," *Sci. Rep.*, vol. 13, no. 1, p. 4400, Mar. 2023, doi: [10.1038/s41598-023-31193-8](https://doi.org/10.1038/s41598-023-31193-8).
- [32] N. Jeffrey, Q. Tan, and J. R. Villar, "A review of anomaly detection strategies to detect threats to cyber-physical systems," *Electronics*, vol. 12, no. 15, p. 3283, Jul. 2023, doi: [10.3390/electronics12153283](https://doi.org/10.3390/electronics12153283).
- [33] M. A. Belay, S. S. Blakseth, A. Rasheed, and P. Salvo Rossi, "Unsupervised anomaly detection for IoT-based multivariate time series: Existing solutions, performance analysis and future directions," *Sensors*, vol. 23, no. 5, p. 2844, Mar. 2023, doi: [10.3390/s23052844](https://doi.org/10.3390/s23052844).
- [34] G. Li and J. J. Jung, "Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges," *Inf. Fusion*, vol. 91, pp. 93–102, Mar. 2023, doi: [10.1016/j.inffus.2022.10.008](https://doi.org/10.1016/j.inffus.2022.10.008).
- [35] B. Kim, M. A. Alawami, E. Kim, S. Oh, J. Park, and H. Kim, "A comparative study of time series anomaly detection models for industrial control systems," *Sensors*, vol. 23, no. 3, p. 1310, Jan. 2023, doi: [10.3390/s23031310](https://doi.org/10.3390/s23031310).
- [36] V. Mishra, B. K. Karumuri, N. M. Gautier, R. Liu, T. N. Hutson, S. L. Vanhoof-Villalba, I. Vlachos, L. Iasemidis, and E. Glasscock, "Scn2a deletion improves survival and brain–heart dynamics in the Kcna1-null mouse model of sudden unexpected death in epilepsy (SUDEP)," *Hum. Mol. Genet.*, vol. 26, no. 11, pp. 2091–2103, Jun. 2017, doi: [10.1093/hmg/ddx104](https://doi.org/10.1093/hmg/ddx104).
- [37] L. Hussain, W. Aziz, J. S. Alowibdi, N. Habib, M. Rafique, S. Saeed, and S. Z. H. Kazmi, "Symbolic time series analysis of electroencephalographic (EEG) epileptic seizure and brain dynamics with eye-open and eye-closed subjects during resting states," *J. Physiol. Anthropol.*, vol. 36, no. 1, p. 21, Dec. 2017, doi: [10.1186/s40101-017-0136-8](https://doi.org/10.1186/s40101-017-0136-8).
- [38] M. M. R. Krishnan, S. V. Sree, D. N. Ghista, E. Y. K. Ng, Swapna, A. P. C. Ang, K.-H. Ng, and J. S. Suri, "Automated diagnosis of cardiac health using recurrence quantification analysis," *J. Mech. Med. Biol.*, vol. 12, no. 4, Sep. 2012, Art. no. 1240014, doi: [10.1142/s0219519412400143](https://doi.org/10.1142/s0219519412400143).
- [39] P. Ponikowski, S. D. Anker, T. P. Chua, R. Szelemej, M. Piepoli, S. Adamopoulos, K. Webb-Peploe, D. Harrington, W. Banasiak, K. Wrabec, and A. J. S. Coats, "Depressed heart rate variability as an independent predictor of death in chronic congestive heart failure secondary to ischemic or idiopathic dilated cardiomyopathy," *Amer. J. Cardiology*, vol. 79, no. 12, pp. 1645–1650, Jun. 1997, doi: [10.1016/s0002-9149\(97\)00215-4](https://doi.org/10.1016/s0002-9149(97)00215-4).
- [40] L. Hussain, I. A. Awan, W. Aziz, S. Saeed, A. Ali, F. Zeeshan, and K. S. Kwak, "Detecting congestive heart failure by extracting multimodal features and employing machine learning techniques," *BioMed Res. Int.*, vol. 2020, pp. 1–19, Feb. 2020, doi: [10.1155/2020/4281243](https://doi.org/10.1155/2020/4281243).
- [41] V. Ticcinelli, T. Stankovski, D. Iatsenko, A. Bernjak, A. E. Bradbury, A. R. Gallagher, P. B. M. Clarkson, P. V. E. McClintock, and A. Stefanovska, "Coherence and coupling functions reveal microvascular impairment in treated hypertension," *Frontiers Physiol.*, vol. 8, p. 749, Oct. 2017, doi: [10.3389/fphys.2017.00749](https://doi.org/10.3389/fphys.2017.00749).
- [42] K. C. Bilchick, B. Fetics, R. Djoukeng, S. G. Fisher, R. D. Fletcher, S. N. Singh, E. Nevo, and R. D. Berger, "Prognostic value of heart rate variability in chronic congestive heart failure (veterans affairs' survival trial of antiarrhythmic therapy in congestive heart failure)," *Amer. J. Cardiology*, vol. 90, no. 1, pp. 24–28, Jul. 2002, doi: [10.1016/s0002-9149\(02\)02380-9](https://doi.org/10.1016/s0002-9149(02)02380-9).
- [43] A. Ferchichi, A. B. Abbes, V. Barra, M. Rhif, and I. R. Farah, "Multi-attention generative adversarial network for multi-step vegetation indices forecasting using multivariate time series," *Eng. Appl. Artif. Intell.*, vol. 128, Feb. 2024, Art. no. 107563, doi: [10.1016/j.engappai.2023.107563](https://doi.org/10.1016/j.engappai.2023.107563).
- [44] X. Luo, R. Jiang, B. Yang, H. Qin, and H. Hu, "Air quality visualization analysis based on multivariate time series data feature extraction," *J. Vis.*, vol. 27, pp. 1–18, 2024, doi: [10.1007/s12650-024-00981-3](https://doi.org/10.1007/s12650-024-00981-3).
- [45] G. G. González, S. M. Tagliafico, A. Fernández, G. G. Sena, J. Acuña, and P. Casas, "One model to find them all deep learning for multivariate time-series anomaly detection in mobile network data," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 2, pp. 1601–1616, Apr. 2024, doi: [10.1109/TNSM.2023.3340146](https://doi.org/10.1109/TNSM.2023.3340146).

- [46] L. Billard, A. Douzal-Chouakria, and S. Y. Samadi, "Exploring dynamic structures in matrix-valued time series via principal component analysis," *Axioms*, vol. 12, no. 6, p. 570, Jun. 2023, doi: [10.3390/axioms12060570](https://doi.org/10.3390/axioms12060570).
- [47] Y. Zheng et al., "Correlation-aware spatial-temporal graph learning for multivariate time-series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, Nov. 2023, doi: [10.1109/TNNLS.2023.3325667](https://doi.org/10.1109/TNNLS.2023.3325667).
- [48] M. Hu, X. Feng, Z. Ji, K. Yan, and S. Zhou, "A novel computational approach for discord search with local recurrence rates in multivariate time series," *Inf. Sci.*, vol. 477, pp. 220–233, Mar. 2019, doi: [10.1016/j.ins.2018.10.047](https://doi.org/10.1016/j.ins.2018.10.047).
- [49] Z. Ji, Y. Wang, K. Yan, X. Xie, Y. Xiang, and J. Huang, "A space-embedding strategy for anomaly detection in multivariate time series," *Exp. Syst. Appl.*, vol. 206, Nov. 2022, Art. no. 117892, doi: [10.1016/j.eswa.2022.117892](https://doi.org/10.1016/j.eswa.2022.117892).
- [50] M. Zheng, J. Man, D. Wang, Y. Chen, Q. Li, and Y. Liu, "Semi-supervised multivariate time series anomaly detection for wind turbines using generator SCADA data," *Rel. Eng. Syst. Saf.*, vol. 235, Jul. 2023, Art. no. 109235, doi: [10.1016/j.ress.2023.109235](https://doi.org/10.1016/j.ress.2023.109235).
- [51] Y. Lian, Y. Geng, and T. Tian, "Anomaly detection method for multivariate time series data of oil and gas stations based on digital twin and MTAD-GAN," *Appl. Sci.*, vol. 13, no. 3, p. 1891, Feb. 2023, doi: [10.3390/app13031891](https://doi.org/10.3390/app13031891).
- [52] C. Tang, L. Xu, B. Yang, Y. Tang, and D. Zhao, "GRU-based interpretable multivariate time series anomaly detection in industrial control system," *Comput. Secur.*, vol. 127, Apr. 2023, Art. no. 103094, doi: [10.1016/j.cose.2023.103094](https://doi.org/10.1016/j.cose.2023.103094).
- [53] R. Debrecey, G. L. Gray, W. Tham, K. Goh, and P. Tang, "The development of embedded audit modules to support continuous monitoring in the electronic commerce environment," *Int. J. Auditing*, vol. 7, no. 2, pp. 169–185, Jul. 2003, doi: [10.1111/1099-1123.00067](https://doi.org/10.1111/1099-1123.00067).
- [54] A. Abubakar, C. F. M. Almeida, and M. Gemignani, "Review of artificial intelligence-based failure detection and diagnosis methods for solar photovoltaic systems," *Machines*, vol. 9, no. 12, p. 328, Dec. 2021, doi: [10.3390/machines9120328](https://doi.org/10.3390/machines9120328).
- [55] D. Jing and H.-B. Chen, "SVM based network intrusion detection for the UNSW-NB15 dataset," in *Proc. IEEE 13th Int. Conf. ASIC (ASICON)*, Oct. 2019, pp. 1–4, doi: [10.1109/ASICON47005.2019.8983598](https://doi.org/10.1109/ASICON47005.2019.8983598).
- [56] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, Y. Zhao, and L. Cui, "Robust detection for network intrusion of industrial IoT based on multi-CNN fusion," *Measurement*, vol. 154, Mar. 2020, Art. no. 107450, doi: [10.1016/j.measurement.2019.107450](https://doi.org/10.1016/j.measurement.2019.107450).
- [57] P. V. Huong, L. D. Thuan, L. T. Hong Van, and D. V. Hung, "Intrusion detection in IoT systems based on deep learning using convolutional neural network," in *Proc. 6th NAFOSTED Conf. Inf. Comput. Sci. (NICS)*, Dec. 2019, pp. 448–453, doi: [10.1109/NICS48868.2019.9023871](https://doi.org/10.1109/NICS48868.2019.9023871).
- [58] M. Almiyani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque, "Deep recurrent neural network for IoT intrusion detection system," *Simul. Model. Pract. Theory*, vol. 101, May 2020, Art. no. 102031, doi: [10.1016/j.simpat.2019.102031](https://doi.org/10.1016/j.simpat.2019.102031).
- [59] T. Kim and W. Pak, "Deep learning-based network intrusion detection using multiple image transformers," *Appl. Sci.*, vol. 13, no. 5, p. 2754, Feb. 2023, doi: [10.3390/app13052754](https://doi.org/10.3390/app13052754).
- [60] Y. Yang, H. Fu, S. Gao, Y. Zhou, and W. Shi, "Intrusion detection: A model based on the improved vision transformer," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 9, p. 4522, Sep. 2022, doi: [10.1002/ett.4522](https://doi.org/10.1002/ett.4522).
- [61] Z. Wu, H. Zhang, P. Wang, and Z. Sun, "RTIDS: A robust transformer-based approach for intrusion detection system," *IEEE Access*, vol. 10, pp. 64375–64387, 2022, doi: [10.1109/ACCESS.2022.3182333](https://doi.org/10.1109/ACCESS.2022.3182333).
- [62] D. S. Smys, D. A. Basar, and D. H. Wang, "Hybrid intrusion detection system for Internet of Things (IoT)," *J. ISMAC*, vol. 2, no. 4, pp. 190–199, Sep. 2020, doi: [10.36548/jismac.2020.4.002](https://doi.org/10.36548/jismac.2020.4.002).
- [63] M. Abdel-Basset, H. Hawash, R. K. Chakraborty, and M. J. Ryan, "Semi-supervised spatiotemporal deep learning for intrusions detection in IoT networks," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12251–12265, Aug. 2021, doi: [10.1109/JIOT.2021.3060878](https://doi.org/10.1109/JIOT.2021.3060878).
- [64] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for IoT applications," *Wireless Pers. Commun.*, vol. 111, no. 4, pp. 2287–2310, Apr. 2020, doi: [10.1007/s11277-019-06986-8](https://doi.org/10.1007/s11277-019-06986-8).
- [65] G. Bovenzi, G. Aceto, D. Ciunzio, V. Persico, and A. Pescapé, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–7, doi: [10.1109/GLOBECOM42002.2020.9348167](https://doi.org/10.1109/GLOBECOM42002.2020.9348167).
- [66] L. Yang and A. Shami, "A transfer learning and optimized CNN based intrusion detection system for Internet of Vehicles," in *Proc. ICC IEEE Int. Conf. Commun.*, May 2022, pp. 2774–2779, doi: [10.1109/ICC45855.2022.9838780](https://doi.org/10.1109/ICC45855.2022.9838780).
- [67] J. Toldinas, A. Venčkauskas, R. Damaševičius, Š. Grigaliūnas, N. Morkevičius, and E. Baranauskas, "A novel approach for network intrusion detection using multistage deep learning image recognition," *Electronics*, vol. 10, no. 15, p. 1854, Aug. 2021, doi: [10.3390/electronics10151854](https://doi.org/10.3390/electronics10151854).
- [68] Y. Xiao and X. Xiao, "An intrusion detection system based on a simplified residual network," *Information*, vol. 10, no. 11, p. 356, Nov. 2019, doi: [10.3390/info10110356](https://doi.org/10.3390/info10110356).
- [69] M. Kodyś, S. Lu, K. W. Fok, and V. L. L. Thing, "Intrusion detection in Internet of Things using convolutional neural networks," in *Proc. 18th Int. Conf. Privacy, Secur. Trust (PST)*, Dec. 2021, pp. 1–10, doi: [10.1109/pst52912.2021.9647828](https://doi.org/10.1109/pst52912.2021.9647828).
- [70] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah, "IoT DoS and DDoS attack detection using ResNet," in *Proc. IEEE 23rd Int. Multitopic Conf. (INMIC)*, Nov. 2020, pp. 1–6, doi: [10.1109/INMIC50486.2020.9318216](https://doi.org/10.1109/INMIC50486.2020.9318216).
- [71] J. Song, B. Li, Y. Wu, Y. Shi, and A. Li, "ReAL: A new ResNet-ALSTM based intrusion detection system for the Internet of Energy," in *Proc. IEEE 45th Conf. Local Comput. Netw. (LCN)*, Nov. 2020, pp. 491–496, doi: [10.1109/LCN48667.2020.9314858](https://doi.org/10.1109/LCN48667.2020.9314858).
- [72] A. Shaikh and P. Gupta, "Real-time intrusion detection based on residual learning through ResNet algorithm," *Int. J. Syst. Assurance Eng. Manag.*, vol. 13, pp. 1–15, Jan. 2022, doi: [10.1007/s13198-021-01558-1](https://doi.org/10.1007/s13198-021-01558-1).
- [73] T. T. Khoei, W. C. Hu, and N. Kaabouch, "Residual convolutional network for detecting attacks on intrusion detection systems in smart grid," in *Proc. IEEE Int. Conf. Electro Inf. Technol. (eIT)*, May 2022, pp. 231–237, doi: [10.1109/eIT53891.2022.9813983](https://doi.org/10.1109/eIT53891.2022.9813983).
- [74] C. U. Om Kumar, S. Marappan, B. Murugesan, and P. M. R. Beulah, "Intrusion detection model for IoT using recurrent kernel convolutional neural network," *Wireless Pers. Commun.*, vol. 129, no. 2, pp. 783–812, Mar. 2023, doi: [10.1007/s11277-022-10155-9](https://doi.org/10.1007/s11277-022-10155-9).
- [75] W. Khan, S. Abidin, M. Arif, M. Ishrat, M. Haleem, A. A. Shaikh, N. A. Farooqui, and S. M. Faisal, "Anomalous node detection in attributed social networks using dual variational autoencoder with generative adversarial networks," *Data Sci. Manage.*, vol. 7, no. 2, pp. 89–98, Jun. 2024, doi: [10.1016/j.dsm.2023.10.005](https://doi.org/10.1016/j.dsm.2023.10.005).
- [76] W. Khan and M. Haroon, "A pilot study and survey on methods for anomaly detection in online social networks," in *Human-Centric Smart Computing (Smart Innovation, Systems and Technologies)*, vol. 316, S. Bhattacharyya, J. S. Banerjee, and M. Köppen, Eds. Singapore: Springer, 2023, pp. 119–128, doi: [10.1007/978-981-19-5403-0_10](https://doi.org/10.1007/978-981-19-5403-0_10).
- [77] W. Khan and M. Haroon, "An efficient framework for anomaly detection in attributed social networks," *Int. J. Inf. Technol.*, vol. 14, no. 6, pp. 3069–3076, Oct. 2022, doi: [10.1007/s41870-022-01044-2](https://doi.org/10.1007/s41870-022-01044-2).
- [78] W. Khan, M. Haroon, A. N. Khan, M. K. Hasan, A. Khan, U. A. Mokhtar, and S. Islam, "DVAEGMM: Dual variational autoencoder with Gaussian mixture model for anomaly detection on attributed networks," *IEEE Access*, vol. 10, pp. 91160–91176, 2022, doi: [10.1109/ACCESS.2022.3201332](https://doi.org/10.1109/ACCESS.2022.3201332).
- [79] W. Khan and M. Haroon, "An unsupervised deep learning ensemble model for anomaly detection in static attributed social networks," *Int. J. Cognit. Comput. Eng.*, vol. 3, pp. 153–160, Jun. 2022, doi: [10.1016/j.ijcce.2022.08.002](https://doi.org/10.1016/j.ijcce.2022.08.002).
- [80] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805v2*.
- [81] R. F. Bikmukhamedov and A. F. Nadeev, "Multi-class network traffic generators and classifiers based on neural networks," in *Proc. Syst. Signals Generating Process. Field Board Commun.*, Mar. 2021, pp. 1–7, doi: [10.1109/IEEECONF51389.2021.9416067](https://doi.org/10.1109/IEEECONF51389.2021.9416067).
- [82] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
- [83] K. He, W. Zhang, X. Zong, and L. Lian, "Network intrusion detection based on feature image and deformable vision transformer classification," *IEEE Access*, vol. 12, pp. 44335–44350, 2024, doi: [10.1109/ACCESS.2024.3376434](https://doi.org/10.1109/ACCESS.2024.3376434).
- [84] H. Nizam, S. Zafar, Z. Lv, F. Wang, and X. Hu, "Real-time deep anomaly detection framework for multivariate time-series data in industrial IoT," *IEEE Sensors J.*, vol. 22, no. 23, pp. 22836–22849, Dec. 2022, doi: [10.1109/JSEN.2022.3211874](https://doi.org/10.1109/JSEN.2022.3211874).

- [85] H. Wang and W. Li, "DDoSTC: A transformer-based network attack detection hybrid mechanism in SDN," *Sensors*, vol. 21, no. 15, p. 5047, Jul. 2021, doi: [10.3390/s21155047](https://doi.org/10.3390/s21155047).
- [86] O. Alkadi, N. Moustafa, B. Turnbull, and K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9463–9472, Jun. 2021, doi: [10.1109/JIOT.2020.2996590](https://doi.org/10.1109/JIOT.2020.2996590).
- [87] W. Villegas-Ch, J. Govea, and A. Jaramillo-Alcazar, "IoT anomaly detection to strengthen cybersecurity in the critical infrastructure of smart cities," *Appl. Sci.*, vol. 13, no. 19, p. 10977, Oct. 2023, doi: [10.3390/app131910977](https://doi.org/10.3390/app131910977).
- [88] Y. Liu, Y. Zhou, K. Yang, and X. Wang, "Unsupervised deep learning for IoT time series," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14285–14306, Aug. 2023, doi: [10.1109/JIOT.2023.3243391](https://doi.org/10.1109/JIOT.2023.3243391).
- [89] E. Muhati and D. Rawat, "Data-driven network anomaly detection with cyber attack and defense visualization," *J. Cybersecurity Privacy*, vol. 4, no. 2, pp. 241–263, Apr. 2024, doi: [10.3390/jcp4020012](https://doi.org/10.3390/jcp4020012).
- [90] S. Ayvaz and K. Alpay, "Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time," *Expert Syst. Appl.*, vol. 173, Jul. 2021, Art. no. 114598, doi: [10.1016/j.eswa.2021.114598](https://doi.org/10.1016/j.eswa.2021.114598).
- [91] I. Amihai, R. Gitzel, A. M. Kotriwala, D. Pareschi, S. Subbiah, and G. Sosale, "An industrial case study using vibration data and machine learning to predict asset health," in *Proc. IEEE 20th Conf. Bus. Inform. (CBI)*, vol. 1, Jul. 2018, pp. 178–185, doi: [10.1109/CBI.2018.00028](https://doi.org/10.1109/CBI.2018.00028).
- [92] F. J. Maseda, I. López, I. Martija, P. Alkorta, A. J. Garrido, and I. Garrido, "Sensors data analysis in supervisory control and data acquisition (SCADA) systems to foresee failures with an undetermined origin," *Sensors*, vol. 21, no. 8, p. 2762, Apr. 2021, doi: [10.3390/s21082762](https://doi.org/10.3390/s21082762).
- [93] Z. Li, Y. Wang, and K.-S. Wang, "Intelligent predictive maintenance for fault diagnosis and prognosis in machine centers: Industry 4.0 scenario," *Adv. Manuf.*, vol. 5, no. 4, pp. 377–387, Dec. 2017, doi: [10.1007/s40436-017-0203-8](https://doi.org/10.1007/s40436-017-0203-8).
- [94] M. Esperon-Miguez, P. John, and I. K. Jennions, "A review of integrated vehicle health management tools for legacy platforms: Challenges and opportunities," *Prog. Aerosp. Sci.*, vol. 56, pp. 19–34, 2013, doi: [10.1016/j.paerosci.2012.04.003](https://doi.org/10.1016/j.paerosci.2012.04.003).
- [95] M. Wadinger and M. Kvasnica, "Adaptable and interpretable framework for anomaly detection in SCADA-based industrial systems," *Exp. Syst. Appl.*, vol. 246, Jul. 2024, Art. no. 123200, doi: [10.1016/j.eswa.2024.123200](https://doi.org/10.1016/j.eswa.2024.123200).
- [96] H. Song, D. Rajan, J. Thiagarajan, and A. Spanias, "Attend and diagnose: Clinical time series analysis using attention models," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2018, vol. 32, no. 1, pp. 1–12, doi: [10.1609/aaai.v32i1.11635](https://doi.org/10.1609/aaai.v32i1.11635).
- [97] J. Xu, Y. Hu, H. Liu, W. Mi, G. Li, J. Guo, and Y. Feng, "A novel multivariable time series prediction model for acute kidney injury in general hospitalization," *Int. J. Med. Informat.*, vol. 161, May 2022, Art. no. 104729, doi: [10.1016/j.ijmedinf.2022.104729](https://doi.org/10.1016/j.ijmedinf.2022.104729).
- [98] H. Harb, A. Mansour, A. Nasser, E. M. Cruz, and I. de la Torre Díez, "A sensor-based data analytics for patient monitoring in connected healthcare applications," *IEEE Sensors J.*, vol. 21, no. 2, pp. 974–984, Jan. 2021, doi: [10.1109/JSEN.2020.2977352](https://doi.org/10.1109/JSEN.2020.2977352).
- [99] P. Ordóñez, T. Oates, M. E. Lombardi, G. Hernandez, K. W. Holmes, J. Fackler, and C. U. Lehmann, "Visualization of multivariate time-series data in a neonatal ICU," *IBM J. Res. Develop.*, vol. 56, no. 5, pp. 7:1–7:12, Sep. 2012, doi: [10.1147/JRD.2012.2200431](https://doi.org/10.1147/JRD.2012.2200431).
- [100] S. Huang, H. Qin, Q. Li, and H. Yuan, "A new method for automated driving image defogging based on improved dark channel prior," *J. Phys.: Conf. Ser.*, vol. 2246, no. 1, Apr. 2022, Art. no. 012018, doi: [10.1088/1742-6596/2246/1/012018](https://doi.org/10.1088/1742-6596/2246/1/012018).
- [101] I. Gitman and B. Ginsburg, "Comparison of batch normalization and weight normalization algorithms for the large-scale image classification," 2017, *arXiv:1709.08145*.
- [102] J. Lei Ba, J. Ryan Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.
- [103] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023, doi: [10.1109/TPAMI.2022.3152247](https://doi.org/10.1109/TPAMI.2022.3152247).

LARAIB SANA is currently pursuing the Ph.D. degree with the Department of Computer Science, Lahore College for Women University, Lahore, Pakistan. Her research interests include developing and optimizing artificial intelligence methods to improve anomaly detection, the IoT, and big data research.



MUHAMMAD MOHSIN NAZIR is currently an Associate Professor with the Department of Computer Science, Lahore College for Women University, Lahore, Pakistan. He is the author of more than 90 research articles. His research interests include security detection, signal and image processing, artificial intelligence, machine learning, and deep learning. He is a member of many research organizations. He has won various prestigious awards and international scholarships.



JING YANG (Graduate Student Member, IEEE) received the B.Eng. degree in navigation technology from Shandong Jiaotong University, in 2022, and the master's degree in data science from the University of Malaya, in 2022. His primary research interests include medical image processing and deep learning.



LAL HUSSAIN received the M.S. degree (Hons.) in communication and networks from Iqra University, Islamabad, Pakistan, in 2012, and the Ph.D. degree from the Department of Computer Science and Information Technology, The University of Azad Jammu and Kashmir, in February 2016. He was a Visiting Ph.D. Researcher with Lancaster University, U.K., for six months under the HEC International Research Initiative Program and under the supervision of Dr. Aneta Stefanovska, Professor of Biomedical Physics, Physics Department, Lancaster University, from 2014 to 2015. He is currently an Assistant Professor with the Department of Computer Science and IT, The University of Azad Jammu and Kashmir, Muzaffarabad, Pakistan. He recently completed a one-year postdoctoral fellowship from the Montefiore Medical Center and the Albert Einstein College of Medicine, Bronx, NY, USA, under the Supervision of Dr. Tim Q. Duong, a Professor and the Vice Chair of MRI Research. He was also with the Duong Laboratory, Stony Brook University, Stony Brook, NY, USA, on different ongoing projects with Dr. Duong, from January 2020 to March 2020. He is the author of more than 80 publications in highly reputed peer-reviewed impact factor journals as a principal author. He completed various funded projects as a PI and Co-PI from Ignite; ICT Pakistan; the University of Jeddah; and Saudi Electronic University, Saudi Arabia. He presented various talks in Pakistan, U.K., India, Peru, Saudi Arabia, and USA. His research interests include developing and optimizing AI tools, including machine learning, deep learning, and neural network algorithms; feature extraction and selection methods; information-theoretic methods; time-frequency representation methods; and cross-frequency coupling to predict disease severity, progression, survival, and recurrence. His area of interests include biomedical signal and image processing problems, including prostate cancer, breast cancer, lung cancer, brain tumors, infectious diseases, including COVID-19 lung infection, viral pneumonia, and bacterial pneumonia, with different modalities, such as MRI, CT, and X-ray, cardiovascular diseases, brain dynamics, and diseases, such as autism spectrum disorder (ASD), attention-deficit/hyperactivity disorder (ADHD), and Alzheimer's disease. Recently, he was ranked in the 2% top world scientists list, in 2021, 2022, and 2023, by Elsevier based on his research record.



YEN-LIN CHEN (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2000 and 2006, respectively. From February 2007 to July 2009, he was an Assistant Professor with the Department of Computer Science and Information Engineering, Asia University, Taichung, Taiwan. From August 2009 to January 2012, he was an Assistant Professor with the Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei, Taiwan, where he was an Associate Professor, from February 2012 to July 2015, and has been a Full Professor, since August 2015. His research results have been published on over 100 journals and conference papers. His research interests include artificial intelligence, intelligent image analytics, embedded systems, pattern recognition, intelligent vehicles, and intelligent transportation systems. He is a fellow of the IET and a member of ACM, IAPR, and IEICE.



CHIN SOON KU received the Ph.D. degree from Universiti Malaya, Malaysia, in 2019. He is currently an Assistant Professor with the Department of Computer Science, Universiti Tunku Abdul Rahman, Malaysia. His research interests include AI techniques (such as genetic algorithm), computer vision, decision support tools, graphical authentication (authentication, picture-based password, and graphical password), machine learning, deep learning, speech processing, natural language processing, and unmanned logistics fleets.

MOHAMMED ALATIYYAH is currently an Assistant Professor with the Department of Computer Science, College of Sciences and Humanities-Aflaj, Prince Sattam Bin Abdulaziz University, Al-Kharj, Saudi Arabia. His research interests include recommender systems, machine learning, deep learning, medical signal processing, and image processing.



SULAIMAN ABDULLAH ALATEYAH received the bachelor's degree in computer science from King Saudi University, Saudi Arabia, in 2004, the master's degree in information technology from University Tenaga Nasional, Malaysia, in 2010, and the Ph.D. degree in computer science from the University of Southampton, U.K., in 2014. He is currently an Assistant Professor of computer science. His works have been published in international conferences and journals. As an academician, his research interests include e-government, e-learning, artificial intelligence, data mining, visualization, tree data structure, data structure and algorithms, and mobile technology and applications.



LIP YEE POR (Senior Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees from Universiti Malaya, Malaysia. He is currently an Associate Professor with the Faculty of Computer Science and Information Technology, Universiti Malaya. His research interests include information security and quality assurance (NEC 2020: 0611), including authentication, graphic passwords, PIN-entry, cryptography, data hiding, steganography, and watermarking. In addition, he specializes in machine learning (NEC 2020: 0613), with expertise in extreme learning machines, support vector machines, deep learning, long-short-term memory, computer vision, and the AIoT.

...