scientific reports

Check for updates

OPEN

A hybrid AI based framework for enhancing security in satellite based IoT networks using high performance computing architecture

Syed Zubair Ahmad[1,2]✉ & Farhan Qamar[1,2]✉

IoT device security has become a major concern as a result of the rapid expansion of the Internet of Things (IoT) and the growing adoption of cloud computing for central monitoring and management. In order to provide centrally managed services each IoT device have to connect to their respective High-Performance Computing (HPC) clouds. The ever increasing deployment of Internet of Things (IoT) devices linked to HPC clouds use various medium such as wired and wireless. The security challenges increases further when these devices communicate over satellite links. This Satellite-Based IoT-HPC Cloud architecture poses new security concerns which exacerbates this problem. An intrusion detection technology integrated in the central cloud is suggested as a potential remedy to monitor and detect aberrant activity within the network in order to allay these worries. However, the enormous amounts of data generated by IoT devices and their constrained computing power dose not allow to implement IDS techniques at source and renders towards typical central Intrusion Detection Systems (IDS) ineffectiveness. Moreover, to protect these systems, powerful intrusion detection techniques are required due to the inherent vulnerabilities of IoT devices and the possible hazards during data transmission.During the course of literature survey it is revealed that the research work has been done to detect few types of attacks by using the old school model of IDS. The computational expensiveness in terms of processing time is also an important parameter to be considered. This work introduces a novel Embedded Hybrid Deep Learning-based intrusion detection technique (EHID) based on embedded hybrid deep learning that is created specifically for IoT devices linked to HPC clouds via satellite connectivity. Two Deep Learning (DL) algorithms are integrated in the proposed method to improve detection abilities with decent accuracy while considering the processing time and number of trainable parameters to detect 14 types of threats. It segregates among the normal and attack traffic. We also modify the conventional IDS approach and propose architectural change to harness the processing power of central server of cloud. This hybrid approach effectively detects threats by harnessing the computing power available at HPC cloud along with leveraging the power of AI. Additionally, the proposed system enables real-time monitoring and detection of intrusions while providing monitoring and management services through HPC using IoT-generated data. Experiments on Edge-IIoTset Cyber Security Dataset of IoT & IIoT indicate improved detection accuracy, reduced false positives, and efficient computational performance.

Keywords Deep learning (DL), Machine learning (ML), IoT, Satellite communication, High-performance computing (HPC) clouds, Embedded hybrid deep learning-based intrusion detection technique (EHID), Intrusion detection systems (IDS)

The Internet of Things (IoT) is an ever-expanding network of interconnected devices, encompassing sensors, actuators, smart homes, and cities. These sectors have collectively formed the foundation for IoT-related industrial

[1]Computer Engineering Department, UET Taxila, Rawalpindi, Punjab 47050, Pakistan. [2]These authors contributed equally: Syed Zubair Ahmad and Farhan Qamar. ✉email: szahmaduet@gmail.com; farhan.qamar@uettaxila.edu.pk

segments. This interconnection generates vast amounts of data, which is processed by high-performance computing (HPC) clouds,which is shaping our world[1]. With the emergence of SpaceX services communication between IoT devices and cloud servers, as well as among the nodes at different sites within the HPC cloud, often relies on satellite communication[2]. By leveraging the wide availability of satellite communication, particularly IoT devices can take advantage of it. However, with the in crease in popularity of IoT and satellite links the concern for the security of IoT devices also increases. Often these IoT devices are deployed in remote and challenging environments, where satellite links are the only viable communication option. This reliance on satellite links, exposes IoT devices to potential cyber-attacks. There are different techniques that are employed to provide resilience against these attacks[3]. To detect and mitigate such attacks, Intrusion Detection Systems (IDS) are commonly used to identify potential attacks in IoT infrastructure, to ensure the provision of reliable services[4,5]. However, IDS for IoT devices have to face unique challenges in this context. The large volume of data generated by these devices, coupled with the limited computing resources available at IoT device. These challenges poses significant hurdles regarding processing power and attack detection. Moreover, IoT devices may be situated in areas with limited connectivity, making it impractical to transmit large volumes of data to a central server for analysis[6,7]. We present Embedded Hybrid Intrusion Detection EHID, a novel solution designed to tackle the challenges associated with intrusion detection in IoT devices as mentioned in[7]. EHID addresses the limitations of limited resources and connectivity, offering efficient and effective intrusion detection capabilities. By implementing EHID, we strive to enhance the security of IoT devices operating in satellite-connected environments, protecting them from potential cyber threats.

In this research we present a Hybrid Deep Learning-based Intrusion Detection Technique specifically tailored around the requirements of IoT devices that are connected to a high-performance computing (HPC) cloud through a satellite link. We are taking advantage of research and dataset in[4]. We proposed a architecture that encompasses on a central HPC cloud (comprising of HPC servers), which are strategically located across different terrestrial locations worldwide. Each IoT device generates data and transmit it to the central HPC cloud for aggregation. The aggregated data is then processed/analyzed by the HPC servers equipped with hardware resources such as Keras/TensorFlow GPU cores. Subsequently, the data undergoes verification by a AI based trained model, enabling real-time intrusion/attack detection and classification of benign and malicious. The model along with its hyper parameters and their effects on processing time required is also considered during the course of our experimentation.

Literature review

We have started our literature review from broader aspects to narrow down to our topic of research in the subsequent sections. The reviewed papers cover a range of topics, including service optimization in cloud-based IoT environments, deepfake detection using deep learning and blockchain, AI and distributed systems in healthcare, data aggregation in IIoT, and machine learning applications in the Internet of Drones. Additionally, there is a focus on nature-inspired algorithms in IoT-based healthcare and the applications of ChatGPT. Together, these studies underscore the growing importance of AI, machine learning, blockchain, and optimization algorithms in enhancing the efficiency, security, and performance of various IoT and healthcare systems. The survey also encompasses the advancement of federated learning and IoT security. By addressing critical challenges such as model poisoning attacks, privacy preservation, irregular user behavior, and energy efficiency. They underscore the necessity for innovative frameworks and architectures that not only enhance the robustness of machine learning models but also ensure the privacy and security of sensitive data in various applications, particularly in healthcare and IoT environments.

Vakili et al.[8] proposes a novel service composition method using grey wolf optimization and MapReduce for cloud-based IoT environments. This approach aims to optimize service selection and allocation to enhance performance and efficiency.

Heidari et al.[9,10] focus on deepfake detection using deep learning methods and blockchain technology. They explore various approaches to detect and mitigate the threat of deepfakes, including federated learning and blockchain-based authentication.

Aminizadeh et al.[11] discusses the opportunities and challenges of AI and distributed systems in improving healthcare service quality. This paper highlights the potential benefits of AI in areas such as diagnosis, treatment planning, and patient monitoring.

Amiri et al.[12] reviews the applications of deep learning in IoT-based bio-and medical informatics. This paper explores the potential of deep learning for tasks such as medical image analysis, wearable device data processing, and personalized healthcare.

Heidari et al.[13–16] explore various applications of AI and IoT, including non-destructive material characterization, data aggregation in industrial IoT, intrusion detection in IoT drones, and machine learning in IoT drones.

Amiri et al.[17] reviews the applications of nature-inspired algorithms in IoT-based healthcare service. This paper highlights the potential of algorithms like genetic algorithms and particle swarm optimization for optimizing healthcare processes.

Heidari et al.[18] provides an overview of ChatGPT, including its components, capabilities, applications, and opportunities. This paper discusses the potential of ChatGPT for various tasks such as natural language generation, translation, and customer service.

Yazdinejad et al in the papers mentioned below, in this section, focuses on federated learning and privacy preservation in IoT and healthcare applications. In[19], a robust privacy-preserving federated learning model is proposed to defend against model poisoning attacks. This study highlights the importance of privacy preservation in federated learning.

In[20], an audit-able privacy-preserving federated learning framework is proposed for electronics in healthcare. This study demonstrates the potential of federated learning in healthcare applications while ensuring privacy preservation.

In[21], a hybrid privacy-preserving federated learning approach is proposed to counter irregular users in next-generation IoT. This study emphasizes the need for robust privacy preservation mechanisms in IoT applications.

In[22], an energy-efficient SDN controller architecture is proposed for IoT networks with blockchain-based security. This study highlights the importance of energy efficiency and security in IoT networks.

Overall, these studies demonstrate the growing importance of federated learning, privacy preservation, and blockchain-based security in IoT and healthcare applications. They also highlight the need for robust and energy-efficient solutions to ensure the integrity and confidentiality of data in these applications. However after having a broad spectrum literature review we focus the background of our work in up coming section.

Background and related work

Security of IoT devices connected to High-Performance Computing (HPC) clouds via satellite links poses a complex challenge but it is a bit simple in a way that it also uses Ethernet protocol. So a comprehensive approach and TCP/IP based analysis is applicable which is deduced from the analysis done in Table 1 along with literature survey, mentioned below. Enhancement of security involves employing the proposed architecture with a embedded hybrid intrusion detection technique with in central server. This DL technique integrates two models. In previous studies, different approaches and multiple AI models have been explored to achieve the same goal. Prasad et al.[23] and Illiashenko et al.[24], suggested the Network Segmentation and Isolation technique. This method requires division of the network into secure zones. The zoning is done on the bases of the criticality and sensitivity of data and devices. By intelligently isolating IoT devices from critical systems the author is able to reduce the potential attack surface.

Satellite link security can be achieved by, implementing the blockchain based encryption and authentication for data which is transmitted over satellite links as mentioned respectively by Cao et al.[3] and Wang et al.[43]. They achieved goal by establishing an infrastructure similar to Virtual Private Networks (VPNs) to secure communication amoung IoT devices and the cloud. Further more, the usage of satellite edge computing, as proposed in[43], is also considered as a promising technique.

Further more another approach was adopted by, Rashid et al.[44], to deployed Host Based Intrusion Detection System (HIDS) agents on every IoT sensor device to deter local threats. The authors conducted detailed analyses on system generated logs, integrety of files, and behavioral pattern deviations. Resultantly generating alerts and responding to the suspicious activities.

In the study conducted by Pandey et al.[45], a Network-Based Intrusion Detection System (NIDS) was implemented at network entry points (e.g., cloud entry) to monitor traffic. It was designed to detect and block known attack patterns, malicious payloads, or abnormal traffic flows in any type of network, whether it was cellular or satellite-connected.

Author(s)	Year	Focus	Deep learning model	Accuracy
Wang et al.[25]	2022	Intrusion detection	CNN	92.16
Sankaran et al.[26]	2023	Anomaly detection, Energy efficiency	RMC-CNN	98.6%
Alabsi et al.[27]	2023	Feature selection, Attack detection	Dual CNN	98.09%
Rahim et al.[28]	2023	Anomaly detection, Face recognition	Logit-boosted CNN	94%
Prasath et al.[29]	2023	DDoS attack defense	Feature optimization, IDS	99.476
Liu et al.[30]	2019	Anomaly detection	GRU, SVDD	96.70%
Rehman et al.[31]	2021	DDoS attack detection	GRU	99.69%
Zhu et al.[32]	2023	Multifeature fusion	Multi-neural network fusion	99.32%
Bokka and Sadasivam[33]	2023	RPL attack detection	GRU	95.51%
Sagar et al.[34]	2023	Intrusion detection	CNN-GRU	Not mentioned%
Banaamah and Ahmad[35]	2022	Intrusion detection	Proposed CNN, LSTM, GRU	99%
Altaf et al.[36]	2023	Intrusion detection	Node-Edge Graph Convolutional Network	97.64%
Altaf et al.[37]	2023	Intrusion detection	Multigraph Neural Network	98.91%
Esmaeili et al.[38]	2023	Malware detection	Graph Neural Network (GNN)	Not mentioned
Alkahtani and Aldhyani[39]	2021	Botnet attack detection	CNN-LSTM	89.64%
Wang and Lu[40]	2020	Anomaly detection	XGBoost, LSTM, Fusion model	98.30%
Ullah et al.[41]	2021	Cyber threat detection	LSTM-CNN	99%
Azumah et al.[42]	2021	Intrusion detection	LSTM	98.00%

Table 1. List of literature survey papers.

Bansla et al.[46], employed machine learning to establish a baseline for IoT device behavior and network traffic. They identified deviations from this baseline as potential intrusion attempts and continuously updated the baseline to adapt to evolving threats, using a hybrid learning approach with anomaly detection.

Kalpana et al.[47], recommended Behavioral Analysis in their work, involving monitoring the behavior of IoT devices and interactions with the cloud to identify unusual patterns, such as excessive data transfer or unauthorized access attempts. Actions were taken based on detected anomalies, such as blocking or alerting.

Dalal et al.[48], leveraged cloud resources to analyze data from IoT devices and network traffic. They implemented advanced analytics to detect sophisticated threats and zero-day attacks.

Pandey et al.[45] used integration of threat intelligence feeds to stay updated on emerging threats. They used threat feeds to enhance detection capabilities and fine-tune intrusion detection rules.

Research gap

During the course of literature survey it is revealed that accuracy becomes better when it is done by AI to detect few types of attacks and limited number of classes. Even the birds eye view of above survey shows that capability of detection of threats in an IDS can be increased by having more types of attack detection. So first gap is to increase the number of types of attacks detection which is 6 in our survey, but it can be increased to 14 types in our experiment. Second a comparative analysis of processing time and accuracy to chose best accuracy in terms of processing time. Third is training a model for real life IoT devices dataset makes it more suitable and in our scenario best possible option by selecting Edge-IIoT dataset. It enables our trained model to detect 10 different types of IoT devices traffic efficiently. For IoT devices a model gives good result only if it is trained on IoT dataset, even if we use multiple types of DL algorithms. This demands the architectural modifications, a large real life dataset with large number of samples and a good combination of DL algorithms. The reason is also depicted in Table 2 of datasets survey.

As obvious from literature survey mostly IDSs are developed by conventional attack detection techniques in early days and later on their performance was increased by the advent of AI based techniques, but even with the AI based techniques the attack detection was limited to 2,3,5 or 6 different types of attacks. As the number of IoT devices connected to their central cloud via satellite increases, it poses the requirement of detection of further more types of attack. As considered by Liu et al.[49] in a satellite-based connected environment, DDoS attacks (UDP, ICMP, TCP, HTTP), MITM attacks, and Port Scanning are more likely due to the unique vulnerabilities and communication protocols associated with satellite networks. Further more Azer et al.[50] identified Backdoor, DDos, Syn-DDos and UDP-DDos attacks for satellite based networks. So all of the above mentioned attacks are catered in our trained model via dataset. Liu et al.[49] and Azer et al.[50] also identified the features such as packet length, header length, SYN, packet time and time between two frames as candidate features for IDS for satellite based communication. The aforementioned features are also catered during the training cycle of our model making our model suitable for detecting threats for IoT devices connected via satellite.

After that a rigorous analysis keeping an eye on processing demand will lead us towards achieving a better result. It is pertinent to mention that no single approach can provide absolute security. A layered, hybrid approach

Author(s)	Dataset	Focus	Deep learning model	Accuracy
Wang et al.[25]	KDD99	5 classes	CNN	92.16%
Sankaran et al.[26]	2023	3 classes	RMC-CNN	98.6%
Alabsi et al.[27]	IoT-Botnet 2020	2 classes	Dual CNN	98.09%
Rahim et al.[28]	AnoML-IoT	2 classes	Logit-boosted CNN	94%
Prasath et al.[29]	BoNeSi-SlowHTTPtest, CICDDoS	2 classes	Feature optimization, IDS, DDoS attack defense, (CNN) and diagonal XG boosting (CNN-DigXG)	99.476%
Liu et al.[30]	KDD Cup99	2 classes	GRU, SVDD	96.70%
Rehman et al.[31]	CICDDoS2019	2 classes	GRU	99.69%
Zhu et al.[32]	Self created	2 classes	Multi-neural network fusion	99.32%
Bokka and Sadasivam[33]	Self created synthetic	2 classes	GRU	95.51%
Sagar et al.[34]	Self created	2 classes	CNN-GRU	Not mentioned
Banaamah & Ahmad[35]	Bot-IoT	2 classes	Proposed CNN, LSTM, GRU	99%
Altaf et al.[36]	UNSW-NB15 and TON-IoT	2 classes	Node-Edge Graph Convolutional Network	97.64%
Altaf et al.[37]	Ton IoT, BoT IoT, NF-Ton IoT, NF-BoT IoT	2 classes	Multigraph Neural Network	98.91%
Esmaeili et al.[38]	NSS_Mirai_Dataset	3 classes	Graph Neural Network (GNN)	Not mentioned
Alkahtani & Aldhyani[39]	N-BaIoT	2 classes	CNN-LSTM	89.64%
Wang & Lu[40]	ADFA-LD	6 classes	XGBoost, LSTM, Fusion model	98.30%
Ullah et al.[41]	CIDDS-01	2 out of 5 classes	LSTM-CNN	99%
Azumah et al.[42]	IoT network intrusion dataset	6 classes	LSTM	98.00%
Our	Edge-IIoTSet	15 classes	CNN	94.76%

Table 2. Survey of literature and datasets to identify research gap keeping in view the number of classes (attacks).

Dataset	Size	Type	Features	Refs.
TON-IoT	2.5 million	Real-world	Telemetry, normal/6 attacks	[51]
NSL-KDD	125,973	Synthetic	Network data, normal/attack	[52]
UNSW-NB15	2.5 million	Real-world	Network data, normal/attack	[52]
IoT-23 Dataset	2.3 million	Real-world	Network data, normal/attack	[53]
CICIDS2017	2.8 million	Synthetic	Network data, normal/attack	[54]
BoT-IoT	3.4 million	Real-world	Network data, normal/4 attacks	[55]
IoT-BDA	4077	Real-world	IoT Botnet, Malware Analysis	[56]
50K-IPD	52,000	Real-world	Malicious URLs	[57]
DNS dataset	88,000	Real-world	Malicious domain names	[58]
ZeuS Tracker	10,000	Real-world	ZeuS malware samples	[59]
Feodo Tracker	10,000	Real-world	Feodo malware samples/feeds	[60]
RanSAP	1495	Real-world	7 Ransomware/5 benign samples	[61]
MBB-IoT	2 million	Real-world	DDos attack, normal/7 attacks	[62]
CICIoT2023	46 million	Real-world	Network attack, normal/7 attacks	[63]
Edge-IIoTSet	1.3 million	Real-world	10 Sensors normal/15 attacks	[4]

Table 3. List of datasets.

S No	IoT device/sensor	No of records
1	Temperature and Humidity	1,629,749
2	Distance Ultrasonic Sensor	1,143,540
3	Water PH Sensor	746,908
4	Heart Rate	165,319
5	Water Level Sensor	2,295,288
6	IR (Infrared) Receiver	1,307,778
7	Sound Sensor	1,512,883
8	Flame Sensor	1,070,196
9	Modbus TCP Server	159,502
10	Soil Moisture Sensor	1,192,777

Table 4. Variety of IoT sensor and number of records.

combining various intrusion detection techniques will improve your chances of detecting and mitigating threats effectively. Regular monitoring, analysis, and adjustments are key to maintaining the security of IoT devices connected to HPC clouds via satellite links.

Challenges and motivation

Challenges

One of the challenges we encountered in our research was the selection of an appropriate dataset. After a thorough evaluation of various available datasets, as mentioned in Table 3, we determined that the Edge-IIoTSet dataset best suited our research needs. This choice became evident even upon a cursory examination of the dataset options. Even the birds eye view of the following available datasets realises us that Edge-IIoTSet has large enough numbers of samples with 10 different sensors data further more it has 15 different classes one benign and 14 different types of attack which makes it most suitable for our experiment out of available datasets.

In related work, specifically, the TON-IoT dataset proposed by Alsaedi et al.[51], a diverse collection of telemetry data was made available, encompassing both normal and attack samples from various scenarios. While this dataset aimed to emulate real-world conditions by including attack sub-categories and network traffic data, it was the Edge-IIoTSet (2022) dataset, introduced by Ferrag et al.[4], that ultimately emerged as the most suitable cybersecurity resource for IoT and IIoT applications. This dataset was tailored to facilitate the development of Intrusion Detection Systems (IDS) for both centralized and distributed applications, including scenarios involving satellite-based communication. Notably, the Edge-IIoTSet dataset incorporated attack scenarios not found in previous datasets. As a result, we opted to utilize the Edge-IIoTSet dataset and its associated research paper as the foundational basis for our research.

Dataset details

The Edge-IIoTset dataset is a comprehensive and realistic cybersecurity dataset specifically designed for IoT and IIoT applications. It is generated from a purpose-built test-bed featuring a diverse range of devices, sensors, protocols, cloud/edge and satellite configurations as desired in research gap subsection.

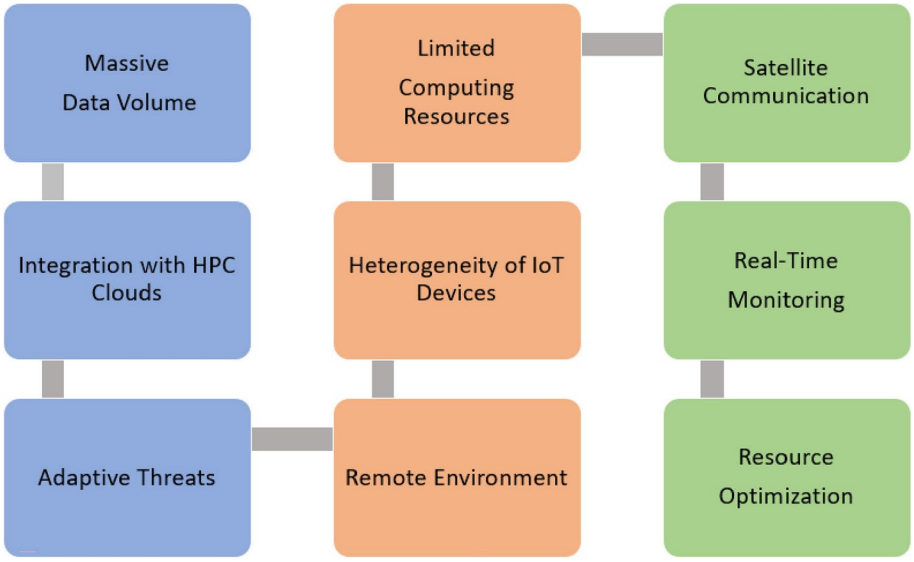

Fig. 1. Challenges.

Class	Count	Focus
Normal	1,363,998	Satellite[50], IoT, Network
DDoS_UDP	121,567	Satellite[50], IoT, Network
DDoS_ICMP	67,939	Satellite[49], IoT, Network
SQL_injection	50,826	IoT , Network
DDoS_TCP	50,062	Satellite[49], IoT, Network
Vulnerability_scanner	50,026	IoT , Network
Password	49,933	IoT , Network
DDoS_HTTP	48,544	Satellite[49], IoT, Network
Uploading	36,807	IoT , Network
Backdoor	24,026	Satellite[50], IoT, Network
Port_Scanning	19,977	IoT, Network
XSS	15,066	IoT, Network
Ransomware	9689	IoT, Network
Fingerprinting	853	IoT, Network
MITM	358	Satellite[49], IoT, Network

Table 5. Dataset split count and attacks focus on related network type.

The IoT data is generated from various IoT devices (more than 10 types) such as Low-cost digital sensors for sensing temperature and humidity, Ultrasonic sensor, Water level detection sensor, pH Sensor Meter, Soil Moisture sensor, Heart Rate Sensor, Flame Sensor, etc.). Furthermore, we identify and analyze over all fourteen attacks including satellite related to IoT and IIoT connectivity protocols. These attacks are categorized into five threats, including, DoS/DDoS attacks, Information gathering, Man in the middle attacks, Injection attacks, and Malware attacks. In addition, we extract features obtained from different literature sources of IDSes for satellite based connectivity. Thus, making it suitable for IoT devices connected to HPC systems via satellite. The details of dataset split count are mentioned in Table 5. The dataset comprises of 1,909,671 samples. The dataset comprises of two major classes i.e Benign and Malicious. Malicious class is categorized into 14 categories based on different attacks types. The detail of each class of dataset along with the attack association type (Satellite/IoT/Network) is provided in Table 5.

Additionally, we have compiled a list of challenges in Fig. 1 inherent to the aforementioned environment, which are outlined below:

- *Massive data volume:* IoT devices generate vast amounts of data, and processing this data efficiently is a significant challenge. Traditional intrusion detection systems (IDS) may struggle to handle the sheer volume of data produced by IoT devices[44].
- *Limited computing resources:* IoT devices typically have limited computing resources, which makes it challenging to implement resource-intensive security measures. Traditional IDS approaches may not be suitable

for these resource-constrained devices. In our research computationally expensive tasks are handed over to central HPC.

- *Satellite communication:* IoT devices Deployed in remote areas rely on satellite communication for connectivity. Satellite links also increases the attack surface. Resulting the potential for chances of eavesdropping and cyber-attacks on data transmitted via satellite[43] as the data communicated on satellite are also in the form of Ethernet packets. Further more in some cases, IoT devices might have limited bandwidth and processing power, which makes it impractical to transmit large volumes of data to a central server for analysis. This limited bandwidth and processing power can affect the ability to detect and respond to local security threats in real-time[23].
- *Integration with HPC cloud:* As mostly IoT devices have to connect to a central server, so Integration of IoT devices with central high performance computing (HPC) cloud to ensure security and efficient data processing is a promising challenge that requires careful design and implementation of proposed architecture. In our research we harness the power of this HPC to provide better security[64].
- *Remote and challenging environments:* IoT devices are often deployed in at different terrestrial locations, remote areas and harsh environments where physical access and provisioning of regular maintenance is difficult. These environmental limitations make it challenging to ensure the security of data and IoT devices.
- *Heterogeneity of IoT devices:* We have different types of IoT devices and IoT sensors with different set of capabilities. Managing and securing these heterogeneous IoT devices is a challenge. That is why we are considering 10 different types of IoT sensors in our research, Incorporated as a part of our dataset[7].
- *Real-time monitoring:* To ensuring real time monitoring of threats detected by intrusion detection system for IoT devices connected is very necessary[43]. This capability paves our way towards identifying and mitigating threats promptly. Central IDS algorithm deployed on central server leads us to achieving this goal in our architecture.
- *Adaptive threats:* The types of cyber attacks on IoT devices are ever evolving and adaptive in nature. Conventional IDS techniques are struggle to keep up with emerging and evergrowing threats/attacks. Resultantly raising the requirement of more advanced and adaptive security measures[24].
- *Resource optimization:* Optimized use of available resources on both sides, on the IoT devices and in the cloud infrastructure is very necessary. The goal is to be achieved while maintaining effective security is a complex task[6]. We propose to placement IDS DL models on central cloud servers to tackle this issue.The paper addresses these challenges by proposing a novel Embedded Hybrid Deep Learning-based Intrusion Detection Technique (EHID) tailored for IoT devices connected to HPC clouds via satellite links. EHID aims to overcome these challenges and enhance the security of IoT devices in such environments.

Motivation

The motivation which proved itself as a guiding star in this research is to secure the IoT device communication. However following are the motivational points that paved our way towards the conclusion of this paper.

- *Growing adoption of IoT and cloud computing:* The increasing adoption of both IoT (Internet of Things) devices and cloud computing technologies has led to a proliferation of IoT deployments. This widespread use of IoT devices and the reliance on cloud infrastructure for data processing have raised significant security concerns.
- *Security concerns for IoT devices:* IoT devices are vulnerable to a wide range of cyber threats due to their ubiquity and often limited security measures. These devices are mostly sensors that collect sensitive environmental data and perform critical functions, making them potentially attractive targets for cyber-attackers[65]. Integrated IDS technique is recommended.
- *Satellite connectivity:* Many IoT deployments, especially in terrestrially remote areas or challenging environments, rely on satellite based connectivity. Not only satellite communication provides wide coverage but also introduces security challenges, which includes MITM, eavesdropping and other cyber-attacks[65]. We consider 14 different types of attacks in our research.
- *Unique challenges for IoT security:* Conventional intrusion detection systems (IDS) are designed for general network based traffic, so the are not specially designed for IoT environments. Due to the bulk data generated by IoT devices and their limited computing resources power[6], the unique vulnerabilities are associated with these devices[46]. An IDS DL based model specially trained on IoT dataset is proposed.
- *Need for real-time monitoring:* Real-time monitoring of an intrusion detection system is essential for identifying and mitigating security threats as they occur. This is most critical in mission critical environments[46]. It can efficiently possible in our proposed scenarios where IoT devices are connected to high-performance computing (HPC) clouds via satellite links.
- *Emerging threat landscape:* Cyber threats are ever evolving and day by day becoming more and more sophisticated. IoT devices are exposed to risk from both known and unknown/ zerod day threats. Thus resultantly raising the requirement to develop advanced and adaptive security measures[24].
- *Resource optimization:* Efficient utilization of the limited computing resources available on IoT devices is required, rather than overloading the IoT device processor as well as ensuring robust security is a complex goal to be achieved[44].
- *Integration with HPC cloud:* Integrating of IoT devices with central HPC cloud provides us with additional computing power for data analysis, attack detection and processing. However, this goal of integration must be achieved securely to protect sensitive IoT data against cyber attack[64].
- *Ensuring IoT resilience:* Enhancement of the security of IoT devices connected to central HPC clouds is critical for ensuring the resilience/protection of these systems against intrusions. This ensures availability of IoT services specially for a mission critical system[23].

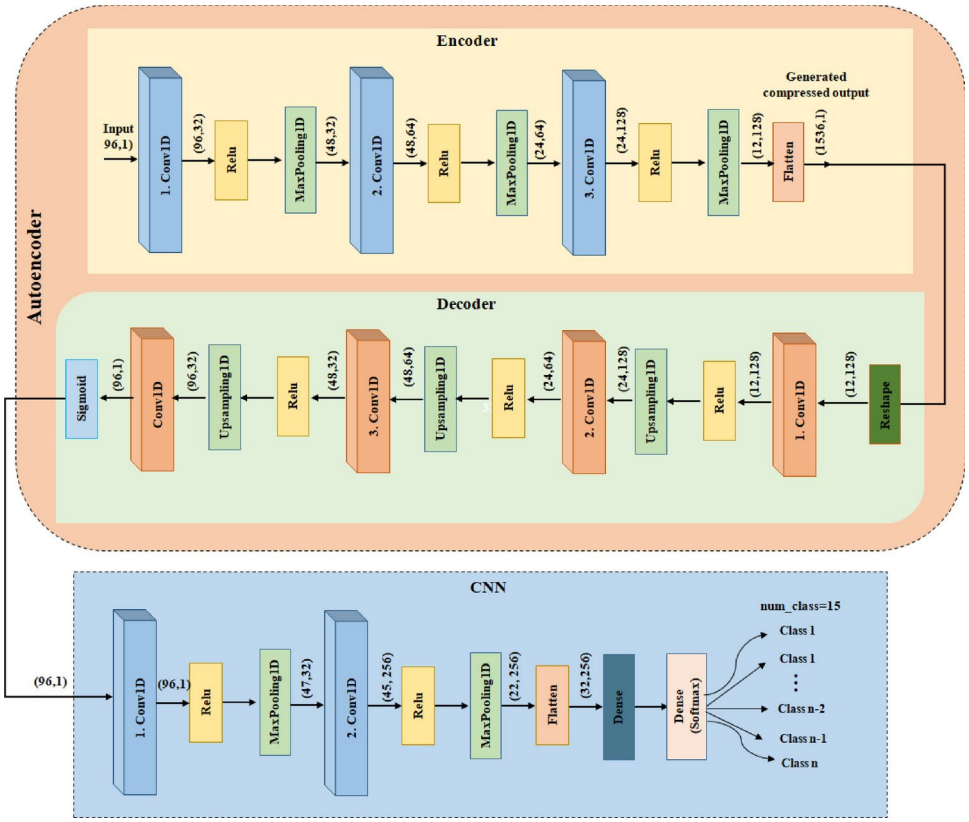

Fig. 2. Architecture of AI models.

- *Practical application:* After a comprehensive literature survey this paper bridges the gap between the security challenges faced by IoT deployments with satellite-connected HPC cloud infrastructure and practical solutions. It sheds a light to propose a novel intrusion detection technique that can be used enhance IoT security in such scenarios.In conclusion, the motivation behind the paper lies in tackling the upcoming security challenges which have targeted the IoT devices specially in satellite-connected HPC cloud environments. In these environments traditional security approaches may fall short. The paper aims to propose an innovative intrusion detection technique (EHID) that takes advantage of deep learning to enhance security, reduce false positives, and improve the overall resilience of IoT systems to cyber threats.

Proposed architecture

In this paper, the authors have suggested a novel architecture that will provide real-time intrusion detection against threats while leveraging the cloud infrastructure's HPC based computational capabilities. The architecture is depicted in Figs. 2 and 3. This architecture exploits the power of both DL models the CNN and the Autoencoders to produce a hybrid intrusion detection system that successfully tackles (with descent accuracy) the specific security issues provided by IoT devices connected to HPC clouds through satellite communications. Our proposed system architecture is depicted in Fig. 1. Here's outline of the proposed architecture:

Sensor layer

This layer consists of the IoT devices that collect data from the environment. The data collected by these devices is then transmitted to the next layer. The IoT devices are responsible for collecting data and training local models using FL. Each IoT device is equipped with a sensor that generates data streams. The data streams are preprocessed and then submitted to their respective cloud for monitoring and management purpose to serve the clients.

Data preprocessing layer

This layer is responsible for preprocessing the data collected from the sensor layer. This includes tasks such as filtering, aggregation, and normalization.To make the raw data consistent and usable, data is cleaned and preprocessed.For accurate analysis, normalize data to bring it to a consistent scale.To reduce noise in the analysis, following steps are performed. The preprocessed data is then transmitted to the next layer.

Intrusion detection layer

This layer performs the task of detecting intrusions/Attacks. Its Architecture is depicted below in Fig. 2. A combination of 1D convolution and autoencoder methods was utilized. The autoencoder had dimensions of

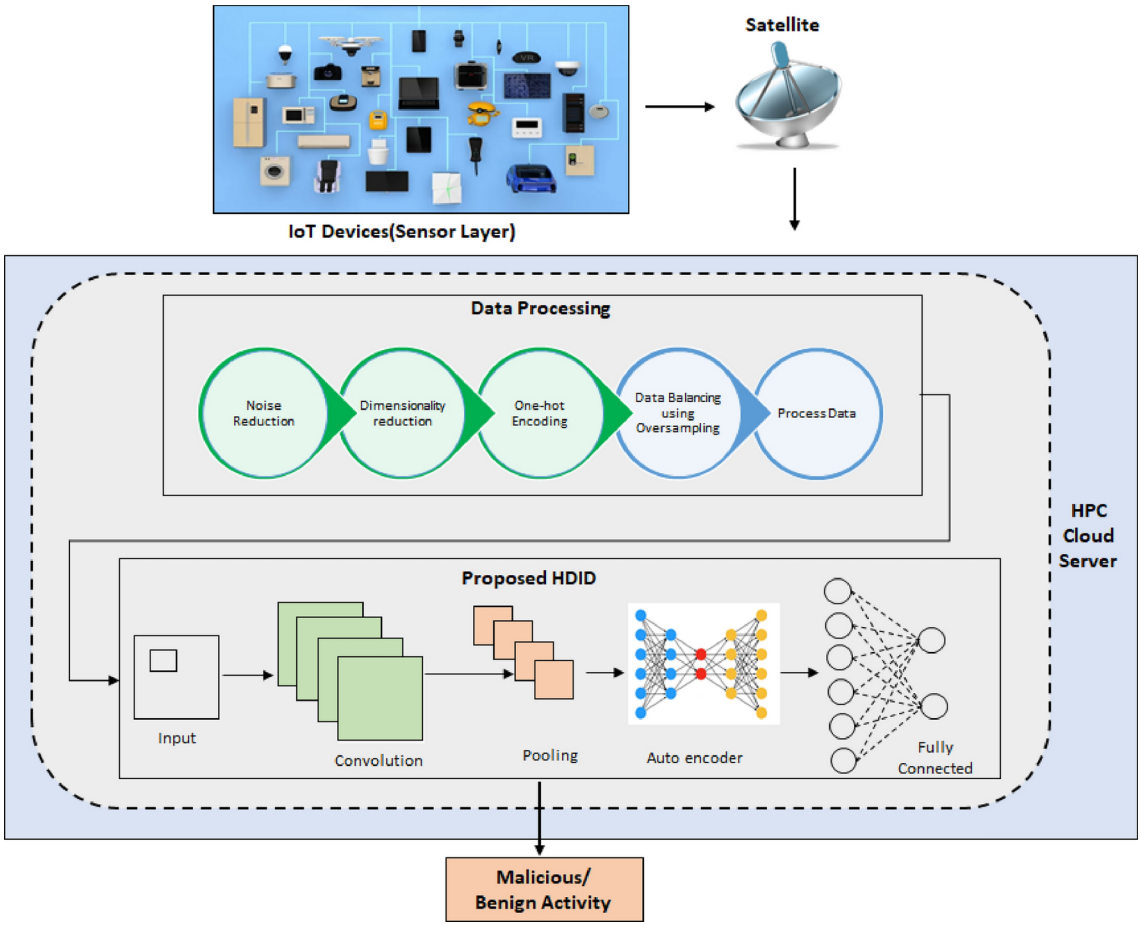

Fig. 3. Proposed Architecture.

32, 64, and 128, coupled with decoder reshaping ranging from 12 to 128. Subsequently, an auto decoder was implemented using dimensions of 128, 64, 32, and 1. Additionally, CNN layers of 32 and 256, along with a CNN Dense layer of 64, were introduced. The ADAM optimizer was chosen based on recommendations from the Edge-IIoT dataset how ever other are also tested in Experimental phase. This is the diagram of our vanilla flavour which is further modified for improvement as mentioned in experiment section. By the application of classification algorithms, benign and malicious instances are distinguished and placed in one of 15 classes.

HPC cloud layer

This layer is responsible for storing and analyzing the data collected from the edge layer. The data is analyzed using a variety of machine learning algorithms to detect intrusions. The local server of cloud is responsible for aggregating the data received from the IoT devices and to train a global model with the help of HPC cloud servers, which provides additional computing resources to support the central cloud to build global model. The cloud is responsible for processing large amounts of data generated by the IoT devices and for providing additional computing power to the central server. The global model is then finaly trained for intrusion detection. The central server also monitors the overall performance of the system and provides alerts in case of any anomalies.

Experimentation

During the implementation of our experiments, we encountered several challenges and limitations. One of the major hurdles was selecting an appropriate dataset that aligns with the requirements of our proposed architecture. In our architecture, IoT devices communicate with their HPC clouds through the TCP/IP protocol to fulfill the essential needs of monitoring and management services provided by the clouds. However, this TCP/IP-based communication is susceptible to eavesdropping by malicious attackers. Therefore, it was crucial for us to obtain a dataset that includes a diverse set of attack data and data from multiple sensors. Fortunately, we were able to find such a dataset as mentioned[4]. Achieving accurate detection of both malicious and benign traffic presents another challenge. Our proposed EHID obtained an accuracy of 94% . The proposed framework is implemented using PyTorch in Anaconda Spyder 5.3.3.

Exp #	Input parameters								Output parameters										
	Epoch (T)	Encoder Conv1D	Decoder reshape	Decoder Conv1D	CNN Conv1D	CNN dense	Opti	Epo	Steps	Time/ epoch (s)	Time/ step (ms)	Train Param	loss	Accu	Val loss	Val accu	MinVal loss	Mean SqErr	VaLMn SqErr
Exp1.1-1	(5)1	32,64,128	12–128	128,64,32,1	32,256	64	Adam	5	6281	894	142	497680	0.265	0.8828	0.1674				
Exp1.1-2	(5)2	32,64,128	12–128	128,64,32,1	32,256	64	Adam	5	6281	892	142	497680	0.1585	0.9261	0.1557				
Exp1.1-3	(5)3	32,64,128	12–128	128,64,32,1	32,256	64	Adam	5	6281	892	142	497680	0.1518	0.9284	0.1769				
Exp1.1-4	(5)4	32,64,128	12–128	128,64,32,1	32,256	64	Adam	5	6281	891	142	497680	0.1537	0.9274	0.18				
Exp1.1-5	(5)5	32,64,128	12–128	128,64,32,1	32,256	64	Adam	5	6281	921	147	497680	0.1606	0.9238	0.1261	**0.9261**	**0.1261**		
Exp1	1	32,64,128	12–128	128,64,32,1	32,256	64	Adam	1	6281	900	143	497680	0.2255	0.9014	0.1703	0.9256	0.1703		
Exp2	1	32,64,128	12–128	128,64,32,1	32,64,128,256	64	Adam	1	6281	981	156	323792	0.1949	0.9127	0.1647	0.9276	0.1647		
Exp3	1	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	1	6281	1028	163	572624	0.1833	0.9191	0.1461	0.9316	0.1461		
Exp4	1(5)	32,64	24–64	64,32,1	32,64,128,256	256	Adam	5	6281	719	114	237520	0.1978	0.9045	0.1657				
Exp4	25)	32,64	24–64	64,32,1	32,64,128,256	256	Adam	5	6281	703	112	237520	0.1583	0.9179	0.1491				
Exp4	3(5)	32,64	24–64	64,32,1	32,64,128,256	256	Adam	5	6281	711	113	237520	0.155	0.9212	0.1746				
Exp4	4(5)	32,64	24–64	64,32,1	32,64,128,256	256	Adam	5	6281	710	113	237520	0.1681	0.9161	0.1682				
Exp4	5(5)	32,64	24–64	64,32,1	32,64,128,256	256	Adam	5	6281	701	112	237520	0.1502	0.923	0.1461	0.9128	0.14613		
Exp5	1(5)	32,64	24–64	64,32,1	32,64,128,256	256	SGD	1	6281	728	116	486352	0.5023	0.836	0.2379	0.8853	0.2379		
Exp5	2(5)	32,64	24–64	64,32,1	32,64,128,256	256	SGD	1	6281	746	119	486352	0.2246	0.8891	0.2212	0.8994	0.2212		
Exp6-1	(5)1	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	5	6281	912	145	572624	0.2771	0.8766	0.1868			0.0097	0.0072
Exp6-2	(5)2	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	5	6281	904	144	572624	0.1625	0.9245	0.162			0.0061	0.0061
Exp6-3	(5)3	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	5	6281	1071	171	572624	0.1649	0.925	0.1528			0.0062	0.0058
Exp6-4	(5)4	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	5	6281	1080	172	572624	0.1451	0.9299	0.1511			0.0055	0.0057
Exp6-5	(5)5	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	5	6281	1042	166	572624	0.1498	0.9289	0.1307	**0.9119**	**0.1307**	0.0057	0.0049
Exp7-1	(12)1	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	1018	161	572624	0.2196	0.9041	0.2184			0.0078	0.0084
Exp7-2	(12)2	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	918	146	572624	0.1664	0.9216	0.1498			0.0063	0.0057
Exp7-3	(12)3	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	912	145	572624	0.1775	0.9183	0.1888			0.0068	0.0073
Exp7-4	(12)4	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	912	146	572624	0.177	0.9181	0.1708			0.0068	0.0065
Exp7-5	(12)5	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	916	146	572624	0.1643	0.9231	0.1775			0.0063	0.0068
Exp7-6	(12)6	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	919	146	572624	0.1705	0.9209	0.1623			0.0065	0.0062
Exp7-7	(12)7	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	1018	161	572624	0.1543	0.9261	0.1705			0.0064	0.0067
Exp7-8	(12)8	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	926	147	572624	0.1637	0.9195	0.1652			0.0063	0.0064
Exp7-9	(12)9	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	924	147	572624	0.1617	0.9204	0.1593			0.0063	0.0062
Exp7-10	(12)10	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	927	148	572624	0.1622	0.9199	0.1738			0.0062	0.0068
Exp7-11	(12)11	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	1140	182	572624	0.1613	0.9201	0.1545			0.006	0.006
Exp7-12	(12)12	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	15	6281	1212	193	572624	0.1559	0.9251	0.1767	0.8869	0.1497	0.006	0.0068
Exp8	(2)1	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	1	6781	1104	162	572624	0.1826	0.9158	0.1269				
Exp8	(2)2	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	1	6781	1141	168	572624	0.1306	0.9371	0.1215	0.9394	0.1215		
Exp9	**(15)10**	**32,64,128**	**12–128**	**128,64,32,1**	**32,64,128,256**	**256**	**Adam**	**15**	**6781**	**1033**	**152**	**572624**	**0.1106**	**0.9476**	**0.1087**	**0.9494**	**0.1087**	**0.0042**	**0.0041**
Exp10	(2)2	32,64,128	12–128	128,64,32,1	32,64,128,256	256	Adam	2	6781	1072	158	572624	0.1783	0.9145	0.1741	0.9201	0.1741	0.0069	0.0067

Table 6. Detailed list of all experimental input parameters and output parameters. Significant values of highest accuracy are in (bold)

Dataset collection

The publicly available dataset "Edge-IIoTset Cyber Security Dataset of IoT & IIoT" containing malicious and benign traffic is collected from Kaggle[66].

Feature selection

To begin, we acquired our dataset in CSV format from Kaggle[66], which served as the starting point for our analysis. Out of the 63 columns in the dataset, we identified 'Attack_label' and 'Attack_type' as the two classifiers essential for our task. We subsequently pruned the dataset by removing columns 'ip.src_host' and 'ip.dst_host' as both are private ip addresses of experimental setup during dataset creation. They were deemed non-contributory to our learning process. Further more, Liu et al.[49] and Azer et al.[50] also identified the features such as packet length, header length, SYN, packet time and time between two frames as candidate features for IDS for satellite based communication. The aforementioned features are also considered during the training cycle of our model making our model suitable for detecting threats for IoT devices connected via satellite.

Data cleaning and preprocessing

After that we proceeded for rows wise NaN value removal, it means that any row with at least one NaN value will be dropped. This step is followed by another step of removal of duplicate values. During the course of this step first of all duplicate rows will be identified based on all columns, then Keep the first occurrence and remove subsequent duplicate rows and after that, the changes will be applied directly to the original Data-Frame.

Subsequently, we executed One-Hot Encoding to convert categorical variables into numerical format. We selected seven columns, namely 'http.request.method', 'http.referer', 'http.request.version', 'dns.qry.name.len', 'mqtt.conack.flags', 'mqtt.protoname', and 'mqtt.topic' as candidates for this transformation. Given the imbalanced nature of certain columns specifically 'Port_Scanning', 'XSS', 'Ransomware', 'Fingerprinting' and 'MITM'-we performed oversampling. To bolster the representation of these minority classes, we increased the number of samples to 25,000 in each of these columns, thereby enabling them to contribute more effectively to the decision-making process.

Hyper parameter analysis

In the realm of model training, we opted for the Adam optimizer with a learning rate of 0.001. Adam, short for Adaptive Moment Estimation, is a popular optimization algorithm employed in the training of machine learning models, particularly deep neural networks. This algorithm combines the advantages of AdaGrad and RMSProp. The chosen learning rate of 0.001 determines the step size taken during each iteration of optimization. It's worth noting that the learning rate significantly impacts the training process and the final model performance. A too-high learning rate can lead to overshooting the optimal solution and instability, while a too-low learning rate might result in slow convergence or getting stuck in local minima. The decision to use a learning rate of 0.001 was reached after careful experimentation to ensure optimal performance.

To facilitate our model training, we implemented a training loop using the TensorFlow machine learning framework. Within this framework, we defined various critical components essential for training a neural network:

- *EPOCHS=5:* This variable defines the number of training epochs, which is the number of times the model will iterate over the entire training dataset.
- *BATCH_SIZE=256:* This variable determines the number of training examples used in each iteration. It impacts memory usage and training speed.
- *call_backs=early_stopping, lr_reduce:* This is a list of callback functions that will be used during training. Callbacks are functions that can be called at different points during training to perform actions such as early stopping or adjusting the learning rate.
- *early_stopping and lr_reduce:* These are likely instances of callback functions, which we have defined earlier in our code. early_stopping is often used to stop training early if a monitored metric (like validation loss) stops improving, while lr_reduce might be used to dynamically adjust the learning rate during training.
- *validation_data and validation_split=0.1:* These arguments allow us to specify a validation dataset or split a portion of the training data for validation.
- *verbose=1:* This argument determines the level of verbosity during training. A value of 1 usually provides progress updates for each epoch.
- *class_weight:* This is a potential argument we used to provide class weights if our dataset is imbalanced. It helps the model to give more importance to underrepresented classes.It's pertinent to mention that we ensured the code for early_stopping and lr_reduce callbacks is defined correctly and that the training and validation data (X_train, y_train, X_test, y_test) are properly prepared before this training loop.

Parameters settings

We are using the train_test_split function to split our balanced dataset into training and testing sets. This function is commonly used in machine learning to divide a dataset into two subsets: one for training the model and the other for evaluating its performance.

Here's a breakdown of the parameters we have used:

- *test_size=0.2:* This parameter specifies the proportion of the dataset that should be allocated for testing. In our case, we have set it to 0.2, meaning 20% of the data will be used for testing.
- *random_state=1:* This parameter sets the random seed for reproducibility. By setting it to 1, we ensure that the same split will be generated every time we run the code with the same dataset and random seed.

- *stratify:* This parameter is used for stratified sampling, which ensures that the distribution of classes in the training and testing sets is similar to the distribution in the original dataset. After this train_test_split operation, we shall have four arrays:
- *Xtrain :* Training set features.
- *Xtest :* Testing set features.
- *ytrain :* Training set labels.
- *ytest :* Testing set labels.Now, we proceed to use these split datasets in our training loop. Lastly, by various experiments we considered adapting the batch size, learning rate, and other hyper-parameters based on our specific problem and dataset characteristics for optimal training performance. Different performance metrics are used to perform a comprehensive evaluation of model's performance from different point of views. Each metric sheds light on a specific aspect of the model's behavior. Depending on the problem domain and specific goals, various metrics can provide valuable insights which paves way to efficient model optimisation.

Performance evaluation metrics

To depict the results obtained by proposed model correctly, following evaluation metrics are used to: Processing Time per Epoch(Pr Time/Epoch), Processing Time per Step(Pr Time/Step), Trainable Parameters (Trainable Param), Loss , recall, F1-score, accuracy, Validation Accuracy(Val accuracy), Validation Loss (Val loss), Minimum Validation Loss(Min val loss), Mean Squared Error(Mean SqErr), Validation Mean Squared Error(ValMean SqErr) precision, ROC-AUC and Number of Attack Classes.

Accuracy Measures the proportion of correctly classified instances out of all instances. This metric is used to evaluate the model's overall performance. Accuracy is the measure of predictions that our proposed model made correctly. Accuracy can be calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Validation Loss (Val loss) Measures the loss of the model on the validation set. This metric is used to evaluate the model's performance on unseen data. Validation loss refers to the loss computed on a validation dataset separated from it that is not used during the entire training process. It is used for the evaluation of the performance of the model on unseen data and to measure and detect over-fitting. The formula for validation loss is the just same as the loss function formula associated to our specific problem. While on the other hand, use of training dataset, to compute the loss on the validation dataset is done as follows.

$$Validation\ Loss = H(p, q) = -\sum_{x \in X} p(x) \log(q(x)) \tag{2}$$

Categorical cross entropy is used in multi-class classification problems where the output belongs to one out of multiple classes.

$$CCE = -\sum y - \text{true} \log(y - \text{pred}) \tag{3}$$

Recall Measures the proportion of true positive instances that are correctly predicted by the model. This metric is used to evaluate the model's ability to detect positive instances. Recall is the capability of a classification model to correctly identify only the pertinent data-points.

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

F1-score Measures the harmonic mean of precision and recall, providing a balanced measure of both. This metric is used to evaluate the model's overall performance. F1 score is a measure of a classification model's accuracy. It varies from 0 to 1, where 1 is best value.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{5}$$

ROC-AUC Measures the area under the receiver operating characteristic curve, which plots the true positive rate against the false positive rate at different thresholds. This metric is used to evaluate the model's ability to distinguish between positive and negative classes.An (ROC) curve is a graphical depiction of a classification model's performance at multiple classification thresholds. AUC measures the whole two dimensional region beneath the entire ROC curve.

Processing Time per Epoch (Pr Time/Epoch) Measures the time taken to complete one epoch of training, which is the time required to process the entire training dataset once. This metric is used to evaluate the computational efficiency of the model.

Processing Time per Step (Pr Time/Step) Measures the time taken to complete one training step, which is the time required to process a single batch of data. This metric is used to evaluate the computational efficiency of the model at a more granular level.

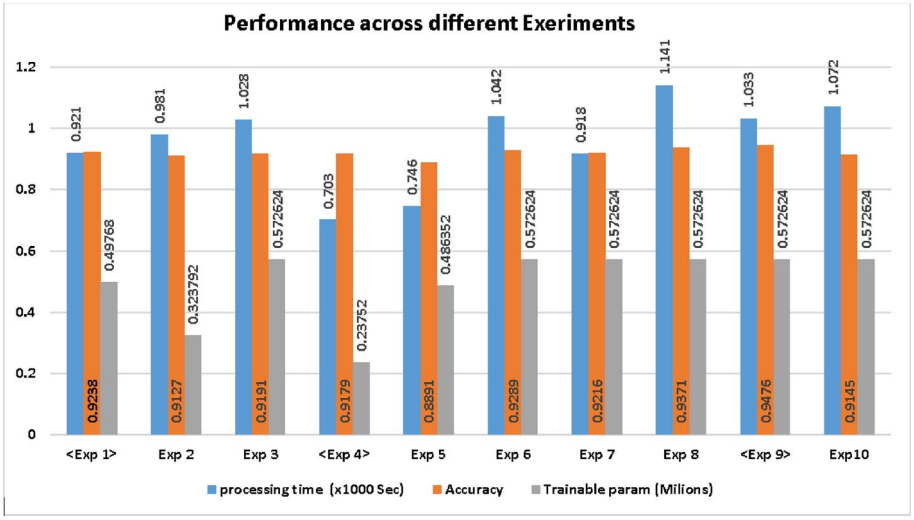

Fig. 4. Performance in terms of Accuracy and Processing Time.

Experiment details	Values
Phases	1 Epoch, 5 Epochs
Techniques used	1D Convolution, Autoencoder
Autoencoder dimensions	32, 64, 128
Decoder reshaping	12–128
Auto decoder dimensions	128, 64, 32, 1
Additional Layers	CNN: 32, 256, CNN Dense: 64
Optimizer	ADAM
Based on recommendations from	Edge-IIoT dataset
Configuration details	Table 6 Input Parameter
Output analysis	Table 6 Output Parameter
Trainable parameters	497,680
Steps	6287
Processing Time	~ 900 s, ~ 921 s (5th Epoch)
Validation Loss (1st Epoch)	0.1703
Validation Loss (5th Epoch, 2nd Phase)	0.1261

Table 7. Experiment no.1 details.

Trainable Parameters (Trainable Param) Represents the number of model parameters that are adjusted during training. This metric is used to evaluate the complexity of the model and its potential to overfit or underfit the data.

Loss Measures the difference between the model's predictions and the actual true labels. This metric is used to evaluate the model's performance during training and is typically minimized to achieve optimal performance.

Validation Accuracy (Val accuracy) Measures the accuracy of the model on the validation set, which is a separate dataset used to evaluate the model's performance during training. This metric is used to evaluate the model's performance on unseen data.

Minimum Validation Loss (Min val loss) Represents the minimum loss achieved by the model on the validation set during training. This metric is used to evaluate the model's best performance on unseen data.

Mean Squared Error (Mean SqErr) Measures the average squared difference between the model's predictions and the actual true labels. This metric is used to evaluate the model's performance, particularly for regression tasks.

Validation Mean Squared Error (Val Mean SqErr) Measures the mean squared error of the model on the validation set. This metric is used to evaluate the model's performance on unseen data, particularly for regression tasks.

Precision Measures the proportion of true positive instances among all predicted positive instances. This metric is used to evaluate the model's ability to avoid false positives.

Number of Attack Classes Represents the number of different attack types or classes that the model is trained to detect. This metric is used to evaluate the model's ability to generalize to different types of attacks.

Results and discussion

During the course of experimentation we have gone through a series of ten experiments which were performed with different parameters on the dataset. The details of all experiments are mentioned in table in Table 6. The variation in these parameters are depicted in table below. The obtained results are also mentioned in the same table so that we can find the experiment with optimum result. But as we go through the series of output we find that different parameters are optimized in different experiments. As shown in Table 6, the proposed Hybrid EHID based IDS techniques outperforms in terms of processing time and validation loss. The EHID based IDS also achieves a higher detection rate and a lower false positive rate. If we focus on our experiment No four in its 2nd out of 5 EPOCH we obtain best processing time (total processing time = 703 s & processing time per epoch = 112 ms) with decent validation loss (val loss = 0.1491)

Ablation study

During the course of experimentation we have gone through a series of 10 experiments. Experiment 1 may be considered as vanilla flavor of our experiments. During these experiments we change the dimensions of autoencoder , CNN layers and epochs. The effect of these changes on processing time , trainable parameters and accuracy are observed/ recorded in tabular format in Table 6 and in graphical format in Fig. 4. Experiment-1 was performed with standard set of autoencoder and CNN dimensions with 1 and 5 epochs. The values of Accuracy, Val loss, Val Accuracy improves as we go from 1 epoch to 5 epochs. Accuracy improves from 90.14 to 92.84. In Experiment-2 changed the shape of CNN Conv1D from (32,256) to (32,64,128,256) keeping the number of epochs = 1. As a effect of this change the accuracy improved from 90.14 to 91.27. In Experiment-3 we kept all parameters same as Experiment-2 but changed the size of CNN Dense layer from 64 to 256, which resultantly increased the accuracy from 91.27 to 91.91 with the increase in number of trainable parameters and processing time. In Experiment-4 we changed the shapes of encoder(32,64,128–32,64), decoder(12,128–24-64), decoder Conv1D(128,64,32,1–64,32,1) and epochs = 5. This set of reduced size slightly reduced the accuracy from 91.91 to 91.79 (Val Accuracy = 92.27) but with great improvement in processing time along with almost half trainable parameters. Further in Experiment-5 we kept all the parameters same as Experiment-4 but just changed the optimiser from Adam to SGD, which drastically reduced the accuracy from 91.79 to 88.91 with increase in trainable parameters and processing time. The Experiment-6 was performed as an extension to Experiment-1 to 3, by keeping all parameters same but changing the shape of CNN Conv1D (32,256–32,64,128,256) and CNN Dense (64–256). This change further improved the accuracy to 92.99 but at the cost of increased processing time and trainable parameters. In the Experiment-7 we kept all the parameters same as in Experiment-6 but increased the epochs from 5 to 15. This change did not resulted in any further increase in Accuracy. Similarly we varied different values of parameters and epochs to find the highest value of accuracy which was found in Experiment-9. The identified best values are listed in Table 10 along with respective values of accuracy and other parameters in Table 9. The minimum processing time is achieved in Experiment-4 with a slightest compromise on Accuracy. The parameters identified for minimum processing time are listed in Table 8.

Standard config experiment 1

The experiment was conducted in two phases: initially with a single epoch, followed by five epochs. In both phases, a combination of 1D convolution and autoencoder techniques was employed. The autoencoder had dimensions of 32, 64, and 128, coupled with decoder reshaping ranging from 12 to 128. Subsequently, an auto decoder was implemented using dimensions of 128, 64, 32, and 1. Additionally, CNN layers of 32 and 256, along with a CNN Dense layer of 64, were introduced. The ADAM optimizer was chosen based on recommendations from the Edge-IIoT dataset. The configuration details are outlined in the "Input Parameter" section of Table 4. Correspondingly, the outcomes of this experiment are documented in the "Output Parameter" section of the same table.

The experiment yielded a total of 497,680 trainable parameters, computed across 6287 steps, and executed within approximately 900 s as shown in Table 4. Notably, during the first epoch, the validation loss was calculated

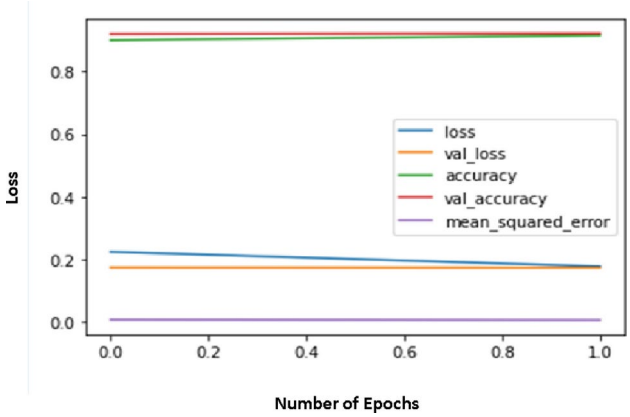

Fig. 5. Effect of Epochs on Loss.

Experiment details	Values
Phases	5 Epochs
Techniques used	1D Convolution, Autoencoder
Autoencoder dimensions	32, 64
Decoder reshaping	12-128
Auto decoder dimensions	64, 32, 1
Additional layers	CNN: 32, 64, 128, 256, CNN Dense: 256
Optimizer	ADAM
Modified on recommendations from	Edge-IIoT dataset
Configuration details	Table 6 Input Parameter
Output analysis	Table 6 Output Parameter
Trainable parameters	237,520
Steps	6,281
Processing Time	~ 703 s, ~ 728 s (5th Epoch)
Validation Loss (1st Epoch)	0.1657
Validation Loss (5th Epoch, 2nd Phase)	0.1461

Table 8. Experiment no.4 details.

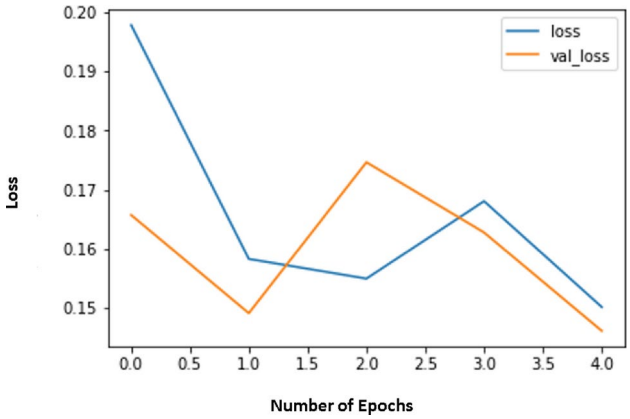

Fig. 6. Graph of Loss and Validation Loss.

at 0.1703 within the 900-s time-frame. With the progression to the fifth epoch in the second phase, this loss exhibited improvement, reaching 0.1261 along with accuracy of 0.9238. However, this enhancement came at a slight trade-off, as the processing time was 921 s.

In essence, the experiment involved a dual-phase approach to refining a model utilizing convolutional and autoencoder techniques. The initial phase served as a benchmark with a single epoch, while the subsequent phase with multiple epochs demonstrated performance improvement depicted in Fig. 5 and Table 7, resulting in optimized validation loss albeit with a modest increase in processing time as depicted in Table 6. The effect is also evident in the heat map drawn in Fig. 7. The model's architecture, parameter details, and outcomes are systematically outlined, aligning with the parameters stipulated by the Edge-IIoT dataset.

Experiment 4 with best processing time

The experiment was conducted in two phases: initially with a single epoch, followed by five epochs as mentioned in Table 8. In both phases, a combination of 1D convolution and autoencoder techniques was employed. The autoencoder had dimensions of 32 and 64 coupled with decoder reshaping ranging from 12 to 128. Subsequently, an auto decoder was implemented using dimensions of 64, 32, and 1. Additionally, CNN layers of 32,64,128 and 256, along with a CNN Dense layer of 256, were introduced. The ADAM optimizer was chosen based on recommendations from the Edge-IIoT dataset. The configuration details are outlined in the "Input Parameter" section of Table 6. Correspondingly, the outcomes of this experiment are documented in the "Output Parameter" section of the same table.

The experiment yielded a total of 237,520 trainable parameters, computed across 6287 steps, and executed within approximately 703 s as shown in Table 8. Notably, during the first epoch, the validation loss was calculated at 0.1657 within the 703-s timeframe. With the progression to the fifth epoch in the second phase, this loss exhibited improvement, reaching 0.1461 as depicted by graph in Fig. 6. However, this enhancement of reduction in processing time (703) came at a slight trade-off of increase in validation loss 0.1451 and decrease in accuracy to 0.9179.

	processing time (x1000 Sec)	Accuracy	Trainable param (Milions)		
Exp 1	0.921	0.9238	0.49768		Max
Exp 2	0.981	0.9127	0.323792		
Exp 3	1.028	0.9191	0.572624		
Exp 4	0.703	0.9179	0.23752		
Exp 5	0.746	0.8891	0.486352		
Exp 6	1.042	0.9289	0.572624		
Exp 7	0.918	0.9216	0.572624		
Exp 8	1.141	0.9371	0.572624		
<Exp 9>	1.033	0.9476	0.572624		
Exp10	1.072	0.9145	0.572624		Min

Fig. 7. Trainable parameters Impacting Highest ValAccuracy.

Experiment details	Values
Phases	15 Epochs
Techniques used	1D Convolution, Autoencoder
Autoencoder dimensions	32, 64, 128
Decoder reshaping	12-128
Auto decoder dimensions	128, 64, 32, 1
Additional layers	CNN: 32, 64, 128, 256, CNN Dense: 256
Optimizer	ADAM
Modifications on recommendations from	Edge-IIoT dataset
Configuration details	Table 6 Input Parameter
Output analysis	Table 6 Output Parameter
Trainable parameters	572,624
Steps	6781
Processing Ttme	~ 1033 s, ~ 1073 s (10th Epoch)
Validation loss (1st Epoch)	0.1258
Validation loss (10th Epoch)	0.1087
Validation loss (15th Epoch)	0.1161

Table 9. Experiment no.9 details.

Parameter details	Values
Loss	0.1106
Accuracy	0.9476
Mean squared error	0.0042
Precision	0.9824
Recall	0.9269
Auc	0.9994
Mean absolute error	0.0083
Mean squared logarithmic error	0.0021
Val loss	0.1087
Val accuracy	0.9494
Val mean squared error	0.0041
Val precision	0.9834
Val recall	0.9288
Val auc	0.9994
Val mean absolute error	0.0082
Val mean squared logarithmic error	0.0020
lr	0.0010

Table 10. Experiment no.9 critical parameters at 10th Epoch.

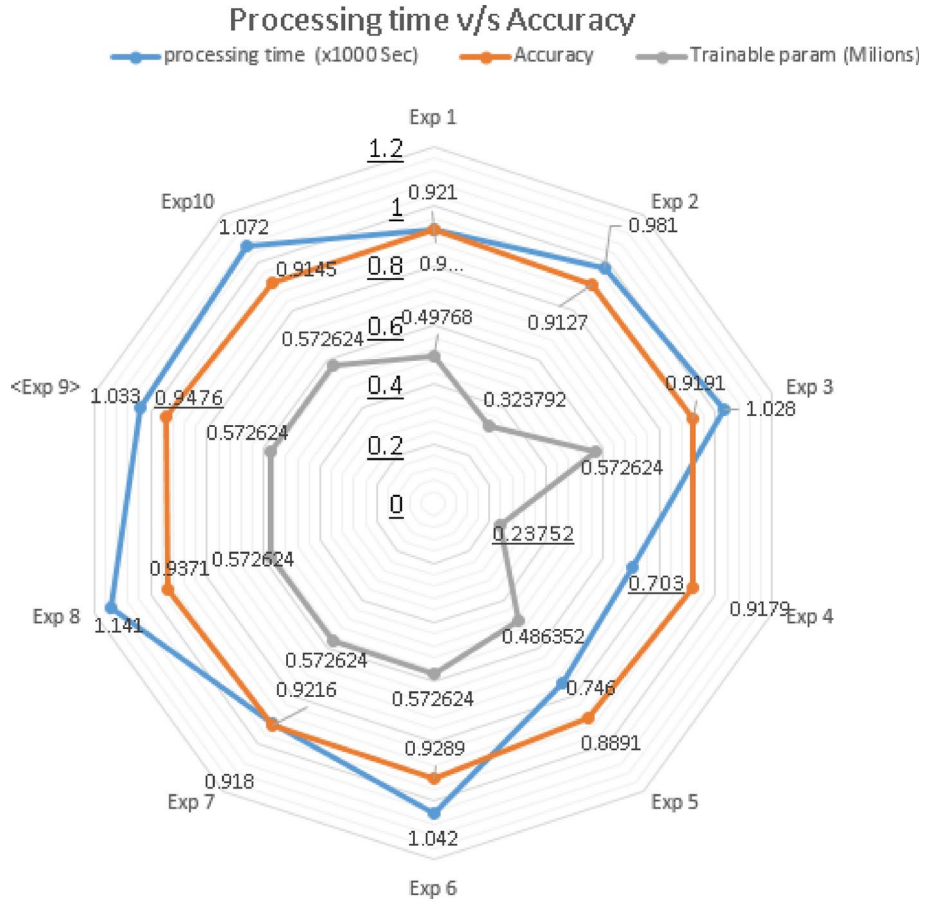

Fig. 8. Trainable parameters impacting Processing Time V/S Accuracy.

In essence, the experiment involved a dual-phase approach to refining a model utilizing convolutional and autoencoder techniques. The initial phase served as a benchmark with a single epoch, while the subsequent phase with multiple epochs demonstrated performance improvement in terms of processing time, resulting in slight increase in validation loss albeit with a modest increase in processing time. The model's architecture, parameter details, and outcomes are systematically outlined, aligning with the parameters stipulated by the Edge-IIoT dataset[4].

Experiment 9 with best accuracy

The experiment was conducted with a single phase of 15 epochs as mentioned in Table 6. In this experiment, a combination of 1D convolution and autoencoder techniques was employed. The autoencoder had dimensions of 32, 64, and 128, coupled with decoder reshaping ranging from 12 to 128. Subsequently, an auto decoder was implemented using dimensions of 128, 64, 32, and 1. Additionally, CNN layers of 32,64,128 and 256, along with a CNN Dense layer of 256, were introduced. The ADAM optimizer was chosen based on recommendations from the Edge-IIoT dataset[4]. The configuration details are outlined in the "Input Parameter" section of Table 6. Correspondingly, the outcomes of this experiment are documented in the "Output Parameter" section of the same table. Improvement in accuracy in all experiments, at the cost of processing time and trainable parameters are depicted by a heat map in Fig. 7.

The experiment yielded a total of 572,624 trainable parameters, computed across 6,781 steps, and executed within approximately 1033 s as shown in Table 9. Notably, during the course of this experiment, the validation loss was calculated at 0.1258 within the 1033-s time-frame. With the progression to the tenth epoch, according to details mentioned in Table 6, in the this phase, this loss exhibited improvement, reaching 0.1087 along with the highest accuracy of 0.9476. However, this enhancement came at a slight trade-off, as the processing time increased to 1073 s. The afore-mentioned statement is further emphasized by the radar chart in Fig. 8 drawn for all experiments having Processing time v/s Accuracy, also catering the number of trainable parameters. It depicts the best accuracy as a function of processing time and trainable parameters for all 10 experiments. The same is further elaborated in Fig. 7 by a heat map. Similarly a birds eye view of Figs. 7 and 8 enables us to conclude that highest accuracy is achieved in Experiment 9, with parameters details mentioned in Table 9 and Critical parameters in Table 10.

In essence, the experiment involved a dual-phase approach to refining a model utilizing convolutional and autoencoder techniques. The initial phase served as a benchmark with a single epoch, while the subsequent

phase with multiple epochs demonstrated performance improvement, resulting in optimized validation loss albeit with a modest increase in processing time. The model's architecture, parameter details, and outcomes are systematically outlined, aligning with the parameters stipulated by the Edge-IIoT dataset.

Conclusions

An architecture with Embeded Hybrid Deep Learning-based Intrusion Detection technique (EHID) for IoT devices connected to an HPC cloud via satellite link is proposed in this paper to ensure the privacy of the data produced by the IoT devices. The suggested system can provide real-time monitoring and intrusion detection as the technique is deployed on centra clud server. The dataset, we used to train and assess the performance of the proposed (EHID) is real life freely available. The technique can be deployed to secure the security and integrity of IoT networks in real life mission critical systems. It shows the potential to use artificial intelligence (DL) approaches to improve the efficiency of Intrusion Detection Technique (IDT) which has been emphasised in this paper. The paper also sheds the light on the difficulties and factors involved in protecting IoT devices using satellite connections.

The Intrusion detection technique can efficiently recognise benign and attack traffic. It can identify 14 differnt potential security attacks by utilising AI technologies like deep learning. Real-time 14 different threat detection capabilities are provided by the IDT, which continuously monitors device activity and satellite network data.

The DL algos are is optimised to minimise resource consumption while keeping strong threat detection capabilities in order to address the specific characteristics of satellite networks, such as low bandwidth, latency, and inconsistent connectivity. The experemental analysis sheds a light on computational expensivenes, and its dependency on trainable parameters. Additionally, the IDT's integration with satellite network considers ethernet as a protocol, as DL algorithms guarantee a smooth and effective functioning.

Scalability analysis

The proposed EHID technique has several built-in mechanisms to enhance scalability, such as placement of HPC at central cloud in IoT infrastructure to support data processing. Deployment of intrusion detection technique on central cloud. It also doesn't require any additional data transmitted over network further more it does not require processing for detection on node or on satellite. However, a more detailed analysis and further optimizations are necessary to ensure it scales effectively with an increasing number of IoT devices and higher data volumes, particularly in satellite networks covering vast geographical areas.

Processing Capability: EHID leverages high-performance computing (HPC) clusters, allowing for distributed and parallel processing of data. This architecture is designed to handle a growing number of devices by distributing computational workloads across multiple nodes. However, as the number of devices increases, the computational demands also grow. To maintain scalability, the servers of EHID must utilize load balancing techniques that dynamically allocate resources based on current network demands. This ensures that no single node becomes a bottleneck, enabling the system to scale efficiently.

Scalability in Satellite Network Environments: In satellite networks, several unique challenges must be addressed for EHID to scale effectively:

Latency and Bandwidth Constraints: Satellite networks often suffer from high latency and limited bandwidth. EHID's centralised processing approach, helps to minimize the amount of data that needs to be transmitted across the network. It doesn't require any additional data to be transmitted over network and it does not require heavy processing on node or on satellite. This reduces latency and conserves bandwidth, ensuring that the system can scale even in challenging satellite network environments.

Geographically Distributed Nodes: The EHID technique is designed to work with geographically distributed nodes, which is a common scenario in satellite networks. The use of centralized processing ensures that data is processed centrally at HPC, reducing the need for long-distance data transmission and enabling more scale able operations.

Adaptability to Variable Network Conditions: EHID is equipped with mechanisms to adapt to variable network conditions, such as changes in data volume or device activity levels. This adaptability ensures that the system can dynamically scale its operations in response to network changes, maintaining high performance across a wide range of scenarios typical of satellite networks. Future work should focus on optimizing communication protocols, adaptive resource management, and real-world testing to fully realize the scalability potential of EHID in these complex environments.

Resilience against advanced persistent threats (APTs)

To fully analyze the effectiveness of the proposed EHIDS in the face of Advanced Persistent Threats (APTs) and sophisticated cyber-attacks, it is necessary to provide a more comprehensive analysis of what threats the system can detect and respond to these types of threats, especially given their complexity and persistence in high-risk environments. The major 14 types of classes of threats that this system can detect are listed in Table 5 of subsection 4.2. Mostly APTs are executed by a sequence of attacks, if this sequence contains any of our 14 types of attacks it will be detected other wise we have to adopt the following mentioned strategies in deployment design to provide resilience.

Integration with Existing Security Measures: EHIDS must work in conjunction with other security measures, such as firewalls and endpoint protection, to create a layered defense strategy. This integration is vital to enhance overall security posture against APTs.

Incident Response: Once an APT is detected, immediate response strategies are crucial. EHIDS can facilitate rapid incident response by providing insights into the nature of the threat and guiding containment efforts. Implementing strict access controls and network segmentation can help limit the spread of an APT within an organization

Regular Audits and Updates: Conducting regular security audits and system updates is vital to patch known vulnerabilities that APTs might exploit. EHIDS should be part of a central security strategy that includes continuous monitoring and improvement of security protocols.

Limitations

Successful deployment of the (EHID) requires regular updates and maintenance procedures, and fostering collaboration between stakeholders for effective reporting and sharing of threat intelligence in dataset. These measures help to address emerging threats and vulnerabilities, ensuring the IDT remains effective in mitigating security risks in future. Further to the the aforementioned item there exists some limitations regarding adaptability of the EHID technique to new types of attacks and the continuous learning of the system to new threats, which may be considered as future research direction. Resilience against APTs is another direction to be considered. During the coarse of our experimentation We have weighed processing vs accuracy how ever measure of processing requirements against the number of IoT devices to be fortified is a task yet to be done in future which will pave its way towards scalability.

Future work

As IoT devices continue to proliferate and satellite connectivity becomes increasingly prevalent, the development and implementation of AI-based IDT for IoT devices connected via satellite links will be a word of mouth in future. This IDT offers a proactive and intelligent security solution to protect IoT networks from 14 different types of threats, ultimately enhancing the overall security posture of IoT deployments over satellite connections. Future work includes exploring the resilience against APTs, scalability and robustness of the proposed system in larger and more complex IoT environments. Incorporation of further more types of sensor data in data set, inclusion of more types of attacks in our dataset. Dataset captured specially from satellite link.Future research directions include optimizing AI algorithms, exploring additional security measures, and addressing scalability challenges for large-scale IoT deployments.

Data availability

The datasets[4] analyzed during the current study are available in the Kaggle repository under the title "Edge-IIoTset Cyber Security Dataset of IoT & IIoT." The dataset can be accessed at https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cyber-security-dataset-of-iot-iiot. However results produced during the course of experiments are mentioned in table of Fig. 4.

Received: 16 June 2024; Accepted: 29 October 2024
Published online: 28 December 2024

References

1. Vimal, S., Suresh, A., Subbulakshmi, P., Pradeepa, S. & Kaliappan, M. Edge computing-based intrusion detection system for smart cities development using IoT in urban areas. *Internet of Things Smart Technol. Sustain. Urban Dev.* 219–237 (2020).
2. Graydon, M. & Parks, L. 'connecting the unconnected': A critical assessment of us satellite internet services. *Med. Cult. Soc.* **42**(2), 260–276 (2020).
3. Cao, S., Dang, S., Zhang, Y., Wang, W. & Cheng, N. A blockchain-based access control and intrusion detection framework for satellite communication systems. *Comput. Commun.* **172**, 216–225 (2021).
4. Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L. & Janicke, H. Edge-IIotset: A new comprehensive realistic cyber security dataset of IoT and IIot applications for centralized and federated learning. *IEEE Access* **10**, 40281–40306 (2022).
5. Ashraf, I. et al. A deep learning-based smart framework for cyber-physical and satellite system security threats detection. *Electronics* **11**(4), 667 (2022).
6. Agrawal, S. et al. Federated learning for intrusion detection system: Concepts, challenges and future directions. *Comput. Commun.* **195**, 346–361 (2022).
7. Talpini, J., Sartori, F., Savi, M. *et al*. A clustering strategy for enhanced fl-based intrusion detection in IoT networks. in *Proceedings of the 15th International Conference on Agents and Artificial Intelligence-(Volume 3)* 152–160 (2023).
8. Vakili, A., Al-Khafaji, H. M. R. & Heidari, A. A new service composition method in the cloud-based internet of things environment using a grey wolf optimization algorithm and mapreduce framework. *Concurr. Comput.: Pract. Exp.* **36**(16), 8091. https://doi.org/10.1002/cpe.8091 (2024).
9. Heidari, A. et al. Deepfake detection using deep learning methods: A systematic and comprehensive review. *Wiley Interdiscip. Rev.: Data Mining Knowl. Discov.* **14**(2), 1520 (2024).
10. Heidari, A. et al. A novel blockchain-based deepfake detection method using federated and deep learning models. *Cognit. Comput.*[SPACE]https://doi.org/10.1007/s12559-024-10255-7 (2024).
11. Aminizadeh, S. et al. Opportunities and challenges of artificial intelligence and distributed systems to improve the quality of healthcare service. *Artif. Intell. Med.* **149**, 102779 (2024).
12. Amiri, Z. et al. The deep learning applications in IoT-based bio-and medical informatics: A systematic literature review. *Neural Comput. Appl.* **36**(11), 5757–5797 (2024).
13. Heidari, A. *et al*. Cloud-based non-destructive characterization. in *Non-Destructive Material Characterization Methods* 727–765 (2024).
14. Heidari, A. et al. A reliable method for data aggregation on the industrial internet of things using a hybrid optimization algorithm and density correlation degree. *Clust. Comput.*[SPACE]https://doi.org/10.1007/s10586-024-04351-4 (2024).
15. Heidari, A. et al. Machine learning applications in internet-of-drones: Systematic review, recent deployments, and open issues. *ACM Comput. Surv.* **55**(12), 1–45 (2023).
16. Heidari, A. et al. A secure intrusion detection platform using blockchain and radial basis function neural networks for internet of drones. *IEEE Internet Things J.* **10**(10), 8445–8454 (2023).
17. Amiri, Z., Heidari, A., Zavvar, M., Navimipour, N. J. & Esmaeilpour, M. The applications of nature-inspired algorithms in internet of things-based healthcare service: A systematic literature review. *Trans. Emerg. Telecommun. Technol.* **35**(6), 4969. https://doi.org/10.1002/ett.4969 (2024).
18. Heidari, A. et al. Everything you wanted to know about chatgpt: Components, capabilities, applications, and opportunities. *Internet Technol. Lett.*[SPACE]https://doi.org/10.1002/itl2.530 (2024).

19. Yazdinejad, A., Parizi, R. M., Dehghantanha, A., Zhang, Q. & Choo, K.-K.R. An energy-efficient SDN controller architecture for IOT networks with blockchain-based security. *IEEE Trans. Serv. Comput.* **13**(4), 625–638 (2020).
20. Yazdinejad, A., Dehghantanha, A. & Srivastava, G. Ap2fl: Auditable privacy-preserving federated learning framework for electronics in healthcare. *IEEE Trans. Consum. Electron.*[SPACE]https://doi.org/10.1109/TCE.2023.3318509 (2023).
21. Yazdinejad, A., Dehghantanha, A., Srivastava, G., Karimipour, H. & Parizi, R. M. Hybrid privacy preserving federated learning against irregular users in next-generation internet of things. *J. Syst. Architect.* **148**, 103088 (2024).
22. Yazdinejad, A., Dehghantanha, A., Karimipour, H., Srivastava, G. & Parizi, R. M. A robust privacy-preserving federated learning model against model poisoning attacks. *IEEE Trans. Inf. Forens. Secur.*[SPACE]https://doi.org/10.1109/TIFS.2024.3420126 (2024).
23. Prasad, G., Chandrika, V. R., Lampe, L. & Vos, G. Enhanced hybrid automatic repeat request scheduling for non-terrestrial IoT networks. arXiv preprint arXiv:2305.07114 (2023)
24. Illiashenko, O., Kharchenko, V., Babeshko, I., Fesenko, H. & Di Giandomenico, F. Security-informed safety analysis of autonomous transport systems considering AI-powered cyberattacks and protection. *Entropy* **25**(8), 1123 (2023).
25. Wang, Y. et al. Network intrusion detection method based on improved CNN in internet of things environment. *Mobile Inf. Syst.* **2022**, 3850582 (2022).
26. Sankaran, K. S. & Kim, B.-H. Deep learning based energy efficient optimal RMC-CNN model for secured data transmission and anomaly detection in industrial iot. *Sustain. Energy Technol. Assess.* **56**(142), 102983 (2023).
27. Alabsi, B. A., Anbar, M. & Rihan, S. D. A. CNN–CNN: Dual convolutional neural network approach for feature selection and attack detection on internet of things networks. *Sensors* **23**(14), 6507 (2023).
28. Rahim, A. et al. Enhancing smart home security: Anomaly detection and face recognition in smart home IoT devices using logit-boosted cnn models. *Sensors* **23**(15), 6979 (2023).
29. Prasath, J., Shyja, V. I., Chandrakanth, P., Kumar, B. K. & Raja Basha, A. An optimal secure defense mechanism for ddos attack in iot network using feature optimization and intrusion detection system. *J. Intell. Fuzzy Syst. (Preprint)* 1–18.
30. Liu, S., Chen, X., Peng, X. & Xiao, R. Network log anomaly detection based on gru and svdd. in *2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)* 1244–1249 (IEEE, 2019).
31. Rehman, S. et al. Diddos: An approach for detection and identification of distributed denial of service (ddos) cyberattacks using gated recurrent units (gru). *Futur. Gener. Comput. Syst.* **118**(147), 453–466 (2021).
32. Zhu, Z. et al. Iot security detection method based on multifeature and multineural network fusion. *Secur. Commun. Netw.* **2023**, 2801421 (2023).
33. Bokka, R. & Sadasivam, T. Securing IoT networks: RPL attack detection with deep learning GRU networks. *Int. J. Recent Eng. Sci.* **10**(2), 13–21 (2023).
34. Sagar, A., Anushkannan, N., Indumathi, G., Muralidhar, N. V., Dhamotharan, K. & Malini, P. Wireless sensor network-based intrusion detection technique using deep learning approach of CNN-GRU. in *2023 8th International Conference on Communication and Electronics Systems (ICCES)* 1147–1152 (IEEE, 2023).
35. Banaamah, A. M. & Ahmad, I. Intrusion detection in IoT using deep learning. *Sensors* **22**(21), 8417 (2022).
36. Altaf, T., Wang, X., Ni, W., Liu, R. P. & Braun, R. Ne-gconv: A lightweight node edge graph convolutional network for intrusion detection. *Comput. Secur.* **130**(156), 103285 (2023).
37. Altaf, T. et al. A new concatenated multigraph neural network for IoT intrusion detection. *Internet Things* **22**(157), 100818 (2023).
38. Esmaeili, B. et al. A GNN-based adversarial internet of things malware detection framework for critical infrastructure: Studying gafgyt, mirai and tsunami campaigns. *IEEE Internet Things J.*[SPACE]https://doi.org/10.1109/JIOT.2023.3298663 (2023).
39. Alkahtani, H. & Aldhyani, T. H. Botnet attack detection by using CNN-LSTM model for internet of things applications. *Secur. Commun. Netw.* **2021**(160), 1–23 (2021).
40. Wang, X. & Lu, X. A host-based anomaly detection framework using XGBboost and LSTM for IoT devices. *Wirel. Commun. Mob. Comput.* **2020**(161), 1–13 (2020).
41. Ullah, I. et al. Software defined network enabled fog-to-things hybrid deep learning driven cyber threat detection system. *Secur. Commun. Netw.* **2021**(162), 1–15 (2021).
42. Azumah, S. W., Elsayed, N., Adewopo, V., Zaghloul, Z. S. & Li, C. A deep LSTM based approach for intrusion detection IoT devices network in smart home. in *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)* 836–841 (IEEE, 2021).
43. Wang, Y., Yang, J., Guo, X. & Qu, Z. Satellite edge computing for the internet of things in aerospace. *Sensors* **19**(20), 4375 (2019).
44. Rashid, M. M. et al. A federated learning-based approach for improving intrusion detection in industrial internet of things networks. *Network* **3**(1), 158–179 (2023).
45. Pandey, B. K. et al. Evaluation of soft computing in intrusion detection for secure social internet of things based on collaborative edge computing. *Soft Comput*[SPACE]https://doi.org/10.1007/s00500-023-08397-1 (2023).
46. Bansal, K. & Singhrova, A. Review on intrusion detection system for IoT/IIoT-brief study. *Multimed. Tools Appl.* **83**(8), 23083–23108 (2023).
47. Kalpana, A. & Waoo, A. A. IoT based smart home cyber-attack detection and defense. *TIJER-Int. Res. J.* **10**(8), 16 (2023).
48. Dalal, S. et al. Next-generation cyber attack prediction for IoT systems: Leveraging multi-class SVM and optimized CHAID decision tree. *J. Cloud Comput.* **12**(1), 1–20 (2023).
49. Li, K., Zhou, H., Tu, Z., Wang, W. & Zhang, H. Distributed network intrusion detection system in satellite-terrestrial integrated networks using federated learning. *IEEE Access* **8**, 214852–214865 (2020).
50. Azar, A. T., Shehab, E., Mattar, A. M., Hameed, I. A. & Elsaid, S. A. Deep learning based hybrid intrusion detection systems to protect satellite networks. *J. Netw. Syst. Manage.* **31**(4), 82 (2023).
51. Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A. & Anwar, A. Ton_iot telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. *IEEE Access* **8**, 165130–165150 (2020).
52. Fuat, T. Analysis of intrusion detection systems in unsw-nb15 and nsl-kdd datasets with machine learning algorithms. *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi* **12**(2), 465–477 (2023).
53. Ahli, A., Raza, A., Akpinar, K. O. & Akpinar, M. Binary and multi-class classification on the iot-23 dataset. in *2023 Advances in Science and Engineering Technology International Conferences (ASET)* 1–7 (IEEE, 2023).
54. Jose, J. & Jose, D. V. Deep learning algorithms for intrusion detection systems in internet of things using cic-ids 2017 dataset. *Int. J. Electr. Comput. Eng. (IJECE)* **13**(1), 1134–1141 (2023).
55. Isma'ila, U. A. & Danyaro, U. D. Muazu: Evaluation on bot-IoT dataset enabled reducing false alarm rate for IoT threats. *KEPES* **21**(3), 490–504 (2023).
56. Trajanovski, T. & Zhang, N. An automated and comprehensive framework for IoT botnet detection and analysis (IoT-BDA). *IEEE Access* **9**, 124360–124383 (2021).
57. Abad, S., Gholamy, H. & Aslani, M. Classification of malicious URLs using machine learning. *Sensors* **23**(18), 7760 (2023).
58. Marques, C., Malta, S. & Magalhães, J. P. Dns dataset for malicious domains detection. *Data Brief* **38**, 107342 (2021).
59. Kazi, M. A., Woodhead, S. & Gan, D. Comparing the performance of supervised machine learning algorithms when used with a manual feature selection process to detect zeus malware. *Int. J. Grid Util. Comput.* **13**(5), 495–504 (2022).
60. Hofbauer, J. Malware beaconing detection with jupyter notebooks. PhD thesis, Technische Hochschule Ingolstadt (2023).
61. Hirano, M., Hodota, R. & Kobayashi, R. Ransap: An open dataset of ransomware storage access patterns for training machine learning models. *Forens. Sci. Int.: Digit. Investig.* **40**, 301314 (2022).

62. Qing, Y., Liu, X. & Du, Y. MBB-IoT: Construction and evaluation of IoT DDoS traffic dataset from a new perspective. *Comput. Mater. Cont.*[SPACE]https://doi.org/10.32604/cmc.2023.039980 (2023).
63. Neto, E. C. P. & Dadkhah, R. Ghorbani: Ciciot2023: A real-time dataset and benchmark for large-scale attacks in IoT environment (2023).
64. Altunay, H. C. & Albayrak, Z. A hybrid CNN+ LSTM based intrusion detection system for industrial IoT networks. *Eng. Sci. Technol. Int. J.* **38**, 101322 (2023).
65. Ashraf, S. N., Manickam, S., Zia, S. S., Abro, A. A., Obaidat, M., Uddin, M., Abdelhaq, M. & Alsaqour, R. Iot empowered smart cybersecurity framework for intrusion detection in internet of drones (2023).
66. kaggle: https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cyber-security-dataset-of-iot-iiot

Author contributions

S.Z. wrote the main manuscript text, and F.Q. supervised and analyzed the paper.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to S.Z.A. or F.Q.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.