



An Intrusion Detection Method for Mobile Networks Utilizing Deep Learning Techniques

Shilpi Agarwal¹, Sudhir Kumar Sharma², Ravi Gupta³

¹Department of ECE & Biomedical Engineering, Jaipur national university, Jaipur, India *
Corresponding

Email: gargshilpi17@gmail.com- ORCID: 0009-0002-4774-8617

² Department of Mathematics, Bannari Amman Institute Of Technology, Sathyamangalam,
Erode, Tamilnadu, India

Email: sudhir.732000@gmail.com - ORCID: 0000-0002-8345-3421

³Department of Electronics and Communication Engineering, Grace College of Engineering,
Thoothukudi, India

Email: raviguptabtp@gmail.com - ORCID: 0000-0002-8150-8046

Abstract: Mobile networks, particularly mobile ad-hoc networks (MANETs), are increasingly deployed in dynamic environments such as military operations, disaster recovery, and remote sensing. However, their decentralized structure, mobility, and open wireless medium make them susceptible to various security threats. This paper presents an intrusion detection method leveraging deep learning techniques to enhance the security of mobile networks. The proposed model uses a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks to effectively learn spatial and temporal patterns of network behavior. By analyzing real-time traffic data, the system can accurately detect anomalies and classify different types of attacks such as blackhole, wormhole, and denial-of-service (DoS). Experimental results demonstrate improved detection accuracy and reduced false positive rates compared to traditional machine learning methods. The deep learning-based approach offers adaptability, scalability, and robustness, making it suitable for real-world deployment in mobile network environments.

Keywords: *Intrusion Detection System (IDS), Mobile Networks, Mobile Ad-Hoc Networks (MANETs), Deep Learning, CNN, LSTM, Cyber security, Anomaly Detection, Wireless Security, Network Threats*

Introduction

The increasing demand for mobile connectivity and wireless communication has led to the rapid proliferation of mobile networks, including Mobile Ad-Hoc Networks (MANETs), Vehicular Ad-Hoc Networks (VANETs), and wireless mesh networks. These mobile networks provide flexible, infrastructure-less communication platforms suitable for dynamic



environments such as disaster recovery zones, military operations, vehicular communication, and remote area surveillance. However, their decentralized nature, open medium, dynamic topology, and lack of centralized monitoring or control make them highly vulnerable to a wide range of cyber attacks and security threats[1][2]. Intrusion Detection Systems (IDS) play a vital role in protecting mobile networks from unauthorized access, malicious activities, and anomalous behavior. Traditional IDS approaches, including rule-based and signature-based systems, are limited in their ability to detect novel or sophisticated attacks and often suffer from high false positive rates. Moreover, the constrained resources of mobile nodes and the inherent volatility of mobile networks pose additional challenges to conventional security frameworks. In recent years, the advancement of artificial intelligence (AI) and machine learning (ML) has brought about significant improvements in intrusion detection capabilities. ML-based IDS have demonstrated improved accuracy and adaptability in detecting known and unknown threats. However, most traditional ML algorithms require manual feature extraction, extensive preprocessing, and often struggle to capture complex patterns in high-dimensional or sequential data. To overcome these limitations, researchers have begun exploring deep learning (DL) techniques as a powerful alternative[3]. Deep learning models are capable of automatic feature extraction and can process complex, nonlinear relationships in large datasets. Models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and autoencoders have shown great promise in analyzing network traffic and detecting various types of intrusions with higher precision and lower false alarm rates[4][5].

The integration of deep learning into IDS design introduces several key advantages. Firstly, DL models can learn both temporal and spatial features from raw network traffic data, enabling them to detect subtle anomalies that may be missed by conventional methods. Secondly, they can be trained on large-scale datasets and adapt to new attack patterns through continuous learning. Thirdly, once deployed, DL-based IDS can operate in near real-time, making them suitable for practical use in mobile networks. Despite these benefits, implementing deep learning in mobile network intrusion detection also comes with its own set of challenges[6][7]. These include the need for high-quality labeled datasets, computational resource constraints on mobile devices, latency issues, and the potential for adversarial attacks targeting the DL models themselves. Addressing these challenges requires careful architectural design, optimization techniques, and consideration of real-world deployment scenarios. The proposed study aims to develop and evaluate an intrusion detection method for mobile networks utilizing deep learning techniques. The primary objective is to design a system capable of accurately identifying both known and unknown attacks while minimizing false positives and computational overhead. The approach involves collecting and preprocessing network traffic data from simulated or real-world mobile environments,



selecting suitable deep learning models such as CNN, LSTM, or hybrid architectures, and training the models to classify traffic into normal or malicious categories. The performance of the proposed system is assessed based on metrics such as detection accuracy, precision, recall, F1-score, and false positive rate[8]. Comparative analysis with conventional machine learning models is conducted to demonstrate the superiority of the deep learning-based approach. In the context of mobile networks, various types of attacks threaten network integrity, availability, and confidentiality. Common threats include blackhole attacks, where a malicious node absorbs all traffic without forwarding it; wormhole attacks, which involve tunneling packets between colluding attackers to disrupt routing; Sybil attacks, where a node claims multiple identities to gain influence; and denial-of-service (DoS) attacks, aimed at overwhelming nodes with traffic[9]. Detecting these threats requires a robust understanding of normal traffic patterns and the ability to distinguish between legitimate deviations and malicious activities. Deep learning's capacity for pattern recognition and anomaly detection makes it a highly suitable candidate for addressing these issues. Additionally, the use of time-series models like LSTM networks is particularly beneficial in mobile networks, where temporal dependencies and event sequences are critical to identifying attack behaviors[10][11].

This research also considers the challenges associated with implementing IDS in resource-constrained mobile environments. Deep learning models, while powerful, are typically computationally intensive and may not be feasible to deploy directly on lightweight mobile nodes. To address this, the study explores lightweight DL models and proposes a centralized or hierarchical IDS framework, where deep learning inference is offloaded to more capable nodes or cloud-based systems while still enabling decentralized detection through localized monitoring. Such a hybrid approach ensures that the benefits of deep learning can be leveraged without compromising the performance or battery life of mobile devices. Furthermore, the adaptability of DL models to evolving threats is supported through periodic retraining and the incorporation of online learning strategies[12].

To validate the effectiveness of the proposed method, simulation environments such as NS2, NS3, or real-world test beds are employed to generate attack scenarios and collect labeled traffic data. The evaluation includes a range of attack types to test the model's generalizability and robustness[13]. The use of benchmark datasets such as NSL-KDD, CICIDS2017, or custom mobile-specific datasets further enhances the credibility of the performance analysis. Additionally, the study incorporates visualization techniques such as confusion matrices, ROC curves, and performance graphs to provide clear insights into model behavior and decision-making processes.



The integration of deep learning into intrusion detection for mobile networks represents a significant step toward more intelligent, adaptive, and resilient cybersecurity solutions. The proposed system not only addresses the limitations of traditional IDS approaches but also aligns with the future trajectory of AI-driven network security. As mobile networks continue to expand in scale and complexity, the need for autonomous and accurate intrusion detection systems will become increasingly critical. This research contributes to that goal by presenting a novel, data-driven, and practical approach to securing mobile communications against evolving cyber threats[14].

Challenges in MANET

Some challenges are brought forth by the components of MANET. A few examples are:

The problem of routing packets between any conglomerate of hubs becomes a testing errand due to the fact that the framework's topology is always hinting at change. Instead of being proactive, most conferences should be set up on open coordination. Due to the autonomous growth of hubs inside the framework, multicast routing has become more difficult as the multicast tree is no longer immobile. Other than the single-bounce correspondence, routes between hubs may have many hops.

Dependability and Security: even if wireless affiliation has constant flaws, an impromptu designated framework has its own security concerns, such scary national giving off packets. Unmistakable affirmation strategies and critical organisation are necessary for the portion of dispersed activity. Data transmission mistakes, convenience-started packet events, the convey notion of the wireless medium, forced wireless transmission runs, and other wireless association features all contribute to unwavering quality concerns.

QOS: Maintaining the clear character of organisational levels in a constantly changing environment will be a challenge. In a MANET, the characteristic of good communications makes it difficult to provide definitive confirmations about the services that a device has access to. For intuitive media organisations to benefit, a Mobile QOS must be used over the usual reserve system.

Interconnection: in addition to communication inside an ad hoc designated system, it is common practice to anticipate interconnection between MANET and established networks, which are often IP based. An obstacle for the friendly portability administration is the coexistence of routing protocols in such a mobile phone.

Improved communication connection capabilities for lean power operation are essential for the improved functioning of lightweight mobile terminals with respect to power consumption. Consideration of energy conservation and power-conscious routing is required.



INTRUSION DETECTION SYSTEMS

An intrusion detection system is a framework-level warning tool. After identifying the security deals made to a system framework, it queries an alert message to an element, like a site security officer, so the substance may take some actions in response to the incursion. A review information collection specialist, an identifier that analyses the review data and provides an output response to the site security officer, and the framework itself are all components of an ID (Moreno 2004). It is important to distinguish between intrusion identification design and intrusion detection tactics while discussing IDS in MANET. Neither the design nor the condition are major factors in the intrusion identification process [15]. In the end, both wired and wireless systems may make advantage of inconsistency and exploitation identification. The main point of difference in application is the selection of review data to include in the approach. But since MANET is so new, most intrusion detection systems in MANET rely on anomaly detection. As contrast to different identification techniques, the majority of material on intrusion detection systems in MANET that the writer reviews is on different types of IDS in MANET. There are a lot of books that gloss over the detecting techniques used. The fact that the architecture is capable of using tactics for detecting anomalies and theft is also mentioned by some. In this way, the various IDS models, rather than the detection algorithms used by the structures, are the focus of the current thesis. At the outset, this section discusses MANET threats and the security role of intrusion detection systems (IDS) in MANET. This is the point at which the requirements for MANET IDS are defined. The possible architectures of intrusion detection systems in MANET are finally examined.

INTRUSION DETECTION SYSTEMS IN MANET

Data security is now facing a scenario where assaults are growing rapidly. Security professionals have very multi-faceted tasks due to the increasing computerisation of attack tools, the constant expansion of assaults, the minimal data needed to breach security, and the progressively increasing complexity. Organising has become crucial for firms nowadays. To achieve this goal and satisfy business requirements, government agencies and private companies have developed their own unique and complex systems, including data networks, fusing innovations, information storage frameworks, various encryption methods, wireless access, web managements, Voice over IP (VoIP), and wireless technologies. Because of the increased usage of system frameworks, the systems have become complicated, and a large section of the workforce enters the organisation via virtual private networks. Client Relationship Management (CRM) and web-based business allow for the same amount of partners to access managements via extranets. The system of high-profile companies and government agencies has been compromised by the attackers. It is now easier than ever for



attackers to damage systems, and they attack with greater expertise than in recent memory [17]. As previously mentioned, the number of system attackers is also growing. Therefore, in today's world, the internet serves as a portal for the dissemination of sensitive information, which in turn brings financial, social, and political entertainment for dissatisfied workers, humiliating partnerships, and even oppressive associations based on terror (confusion)

TYPES OF ATTACKS

(a) Denial of Service Attack: This attack expects to attacks the ease of access of a node or the whole system. On the off chance that the attack is fruitful the managements won't be accessible. The aggressor for the most part utilizes radio flag sticking and the battery depletion technique.

(b) Impersonation: If the verification instrument is not legitimately executed a malicious system can go about as a bona fide node and screen the system activity. It can likewise send fake directing packets, and access some private data.

(c) Eavesdropping: This is a uninvolvement attacks. The node basically watches the classified data. This data can be shortly employed by the noxious node. The anonymity data like area, open key, privatekey, secret-key and so forth can be fetched by eavesdropper.

(d) Routing Attacks: The malignant node make routing managements an objective since it is an imperative management in MANETs. There are two flavors to these directing attacks. One is attacks on directing convention and another is attacks on packet sending or conveyance component. The first is gone for obstructing the engendering of routing data to a node .

(e) Black-hole Attack: In this attack, an attacker publicizes a zero metric for all goals making all nodes about it course packets near it. A hateful node directs fake routing data, asserting that it has an perfect course and makes additional great nodes course evidence packets finished the vindictive one. A vindictive node drops all packets that it gets rather than characteristically distribution those packets

Misuse-Based Intrusion Detection Networks

Additionally, signature-based frameworks that recognise security attempts to breach the framework in unique ways are seen as misuse-based frameworks. The intrusion detection system (IDS) stores known intrusion outlines, which are then compared with framework exercises. In this connection, the IDS investigator is only alerted to a warning if a known intrusion meets the typical for a framework. The creation of signatures that properly coordinate the properties of a specific incursion is necessary to prevent false positives.



The components of the protocol or the content of the system's movement are matched in an NIDS by a specific signature. If the NIDS detects motion that corresponds to the signature, an alert will be triggered. Among the several ways to identify known threats, signature detection is the most precise. Always be on the lookout for signs of an intrusion when a signature is involved. Having said that, a unique signature identified almost every malicious action. Consequently, an intrusion detection system (IDS) using signature discovery can find the majority of malicious activity. Certain types of assaults have been identified and described in the context of signature finding in the past[19, 20]. Be that as it may, many resources may identify these little choices. For instance, signature detection is not without its limitations. It can't figure out what drives signature-matching actions. In other words, it sent off alarms even if the motion could be completely normal. On a frequent basis, motion almost seems to be suspicious behaviour. Consequently, false positives are likely to be generated by forward NIDSs that use signature discovery. Signature detection requires attack history in order to generate accurate signatures.

Because of this fact, intrusion detection systems that use signature identification are created. Because signature identification is just a tool for keeping an eye out for careless, unseen assaults or ones that don't have a precise signature. Changing only one component might sometimes render IDS useless against assaults. The signature base will evolve at the scheduled period because new attacks need new signatures and because more information about vulnerabilities is available. The intrusion detection system can only detect intrusions if each packet is compared to each signature. The framework's computing cost increases as a result of the increased transfer speed. When the amount of data sent exceeds the capabilities of the intrusion detection system, packets are either dropped or miss. Under these circumstances, the likelihood of false negatives is high. Intruder detection systems that focus on abuse have numerous prominent approaches. The following are descriptions of a few of them:

Anomaly-based IDS

These detection systems find abuse by monitoring a standard over time and preparing themselves to react when designs deviate from the norm. New structures for anomaly detection are emerging. Using a peculiarity detection framework at the application level allows for end-user activity screening. Inconsistency detection IDS compiles a set of data from the client's system activity. Next, this typical dataset is taken into account as what is typically used. When the client deviates from the typical example, a warning is sent out. The inconsistency detection IDS would alert the system administrator if a client had been logging onto a system during application hours for many months and then suddenly had a surge of logins at 3:00 a.m. The usual or anticipated behaviour of a framework may be shown by



irregularity detection frameworks, which attempt to find departures from the ordinary and hence reveal an achievable intrusion attempt.references [7] and [11]. Client or program behaviour in a framework may be screened using host-based peculiarity detection. The NIDES framework is one that has worked on creating user profiles using anomaly detection methods, such as monitoring program and typing speeds. The adhoc framework developed by In is one example of a systems-based anomaly detection framework. Use the source and destination IP addresses, ports, and the number of associations during a specified duration to typically display the packet stream. In order to find unusual system packet content behaviour on the 1999 DARPA IDS dataset, late system based peculiarity identification tactics employed include building a support vector and using Mahalanob, a distinct algorithm. Frameworks for detecting anomalies have a top-down bench signature approach that has been proposed in. In order for it to assess the efficacy of various frameworks for detecting inconsistencies[18]

Literature Review

Author(s)	Year	Focus Area	Techniques/Models Used	Datasets Used	Key Contributions / Results
E. Padmalatha et al	2025	Network Anomaly Detection	Ensemble Learning	Not specified	Combined multiple classifiers for improved anomaly detection; focused on adaptability and detection of known/unknown threats.
Jyoti Saini et al	2024	IDS in Wireless Sensor Networks (WSNs)	Supervised, Unsupervised, Semi-supervised ML	Not specified	Survey of ML methods in WSNs; comparative analysis, privacy concerns, and future trends discussed.
Sravanthi Dontu et al	2024	Cyberattack Detection in 5G	DRF Optimization, DBRF, CNN	5G-NIDD	Achieved 95.62% accuracy, 98.32% precision using optimized DL model for 5G cyberattack classification.
Salahaldeen Duraibi et al	2024	IoT Cybersecurity	IMFO for FS, LSTM-DSSAE, DTOA	Not specified	Proposed IMFOHDL-ID model for IoT IDS; highlighted improved detection and performance through hybrid DL.
Sangeetha V et al	2024	Real-time Anomaly Detection	LSTM + XGBoost	NF-BoT-IoT	Achieved 99% detection accuracy using hybrid model suitable for CPS/IoT



					intrusion detection.
Apoorva Joshi et al	2024	Software-Defined Security for 5G	CNN + Neural Architecture Search	Not specified	SDS integrated CNN-based IDS achieving 96.4% accuracy; effective on encrypted and abnormal traffic.
Shaoqiang Wang et al	2024	Vehicular IDS	MobileNetV3	Car-Hacking, CICIDS-2017	Lightweight IDS with 100% accuracy (Car-Hacking), 99.98% (CICIDS-2017); suitable for real-time vehicular systems.
Velpula Nithin Krishna et al	2024	Malware Detection	KNN, DT, XGBoost, ANN, Apriori	NSL-KDD	Evaluated various ML classifiers for malware detection and association rule mining.
M. Bommy et al	2023	IDS in MANETs	MLP, SVM, Random Forest, RL-inspired model	Not specified	MLP-based IDS for MANETs with adaptive RL component; aimed at real-time attack detection.
Mohammed Aljebreen et al	2023	DDoS Attack Detection in 5G	MEOA, LSTM, TSA	Benchmark Dataset	MEOADL-ADC method achieved 97.60% accuracy using hybrid DL with optimized hyperparameters.
Abdulraqeb Ahammadi et al	2022	AI in 5G/6G Security	AI-powered Intrusion Detection	Not specified	Surveyed AI techniques enhancing security in 5G/6G; future directions in physical/network-level defenses.

METHODOLOGY

This discussion does not aim to comprehensively cover the entire scope of Intrusion Detection Systems (IDSs) and machine learning. Instead, it focuses specifically on those aspects considered critically important to the present study and necessary to address the research questions. Primarily, the current literature on IDS will be referenced as the foundation for answering the core research question. Although the number of relevant studies may be limited, an effort will be made to select the most pertinent and meaningful ones[12]. In addition to IDS-related literature, research on machine learning techniques will also be reviewed. Within the broader machine learning literature, several relevant concepts and



perspectives will be considered. These will be critically examined and incorporated into the study. The structural framework of the proposed system is illustrated in Figures 1 and 2.

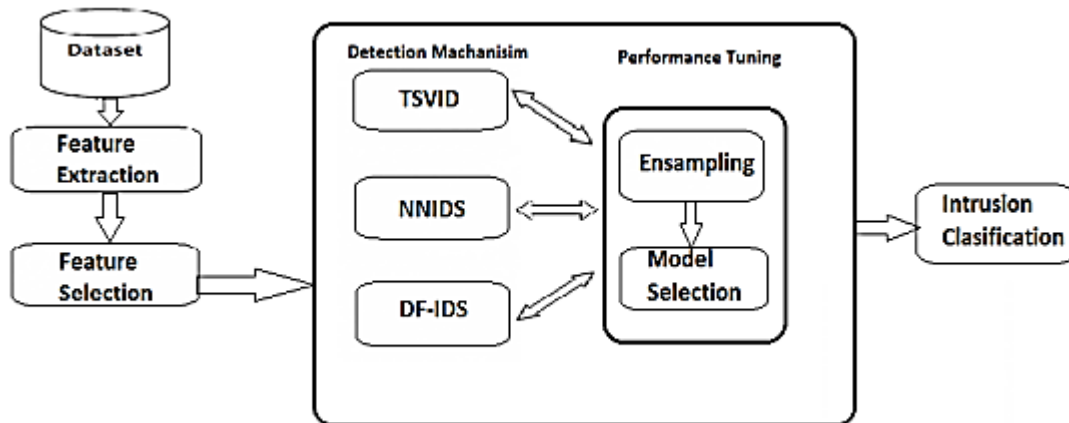


Figure 1 Intrusion Detection Systems Architecture Diagram

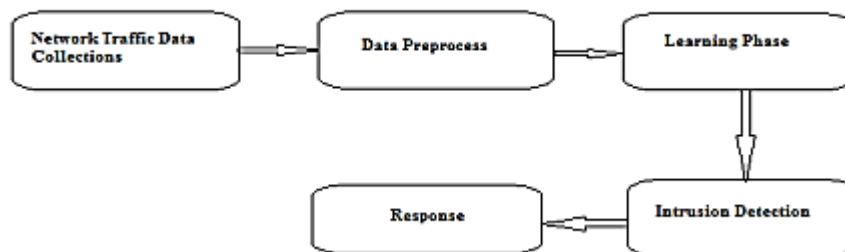


Figure 2 Stages in IDS

Three outstanding machine learning systems will be developed, keeping in mind the end goal of assessing distinct machine learning methodologies and how they interact with IDS input. An approach from a neural systems (NNIS) point of view will constitute the first. Support vector machine (TSVID) will form the foundation of the second system. An additional system based on fuzzy approach (DF-IDF) will be put in place. As will be shown later, these distinct approaches to learning are necessary for achieving noticeable results. These conflicting outcomes will be assessed and separated from the current makeup. Documented details on the progression of events were necessary for the execution. This provides insight into the manual process by which security experts organised alerts. Using the dedication of two systems, the KDD dataset verified the warnings as either common direct or scene. In order to prepare and evaluate the machine learning computation, this stamped information will be used. Information about logs will be a part of the training set. During the course of the test, we will determine the optimal log data fraction and the best way to use the three specific



strategies mentioned before. For intrusion detection systems to achieve their full potential, it is essential that both the number of false positives and the number of false negatives be reduced. As a result, precision, precision, and review are critical metrics to evaluate.

DEEP LEARNING – A DETAILED THEORETICAL PERSPECTIVE

Deep learning is a subset of machine learning that mimics the workings of the human brain in processing data and creating patterns for use in decision-making. It is based on artificial neural networks (ANNs) with multiple layers—hence the term “deep.” While traditional machine learning models depend heavily on human intervention for feature extraction and data pre-processing, deep learning models can automatically learn representations from raw data. This automation makes deep learning particularly powerful for tasks involving large and complex datasets, such as image recognition, natural language processing, and, more recently, network intrusion detection.

At its core, a deep learning model consists of an input layer, several hidden layers, and an output layer. Each layer is composed of nodes, or neurons, which simulate biological neurons. These nodes are interconnected, and each connection has an associated weight. During the training process, the model adjusts these weights using optimization algorithms like stochastic gradient descent to minimize error. The layers perform transformations on the data, allowing the network to learn intricate patterns and hierarchical feature representations. For example, in a deep neural network used for image classification, the early layers might detect edges and textures, while deeper layers detect more abstract patterns like shapes or objects.

The most commonly used deep learning architectures include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), Autoencoders, and more recently, Transformers. Each of these architectures is designed to handle specific types of data and tasks. CNNs, for instance, are especially effective at processing grid-like data such as images and have been adapted for analyzing network traffic in cybersecurity. RNNs and LSTMs, on the other hand, are tailored for sequential data, making them suitable for time-series analysis in mobile networks where the temporal behavior of traffic is crucial for identifying anomalies.

Deep learning has gained prominence in the field of cybersecurity due to its ability to automatically learn complex attack patterns without the need for handcrafted features. Intrusion Detection Systems (IDS), which are designed to monitor network traffic for malicious activities, benefit significantly from this capability. Traditional IDS approaches, such as rule-based or signature-based systems, are limited in detecting novel or previously unseen attacks. They rely on known attack signatures and static rules, which makes them ineffective against evolving threats. Deep learning addresses this limitation by learning from historical data and identifying deviations from normal behavior, even if the specific attack signature is not present in the training set.



In the context of mobile networks, such as Mobile Ad-Hoc Networks (MANETs) and wireless sensor networks (WSNs), the challenges of intrusion detection are even more pronounced. These networks are dynamic, decentralized, and operate over unreliable communication channels. Nodes in a mobile network often have limited computational and power resources, which further complicates the deployment of traditional IDS. Deep learning provides a scalable and intelligent solution for mobile network intrusion detection due to its adaptability and automation capabilities. A deep learning-based IDS can be trained to detect a wide range of attacks, including denial-of-service (DoS), blackhole, wormhole, Sybil, and spoofing attacks. By analyzing network traffic patterns and node behavior over time, it can identify both known and unknown threats with high accuracy.

One of the most effective deep learning models used in intrusion detection is the LSTM network. LSTM is a type of RNN capable of learning long-term dependencies in data, which is essential for analyzing time-based patterns in mobile network traffic. For instance, in a DoS attack, a burst of unusually high traffic might be detected over a short period. An LSTM model can learn the normal sequence of packet flows and identify this abnormal spike as a potential threat. Similarly, autoencoders can be used to learn compact representations of normal traffic data. When new data deviates significantly from this learned representation, it is flagged as anomalous.

Despite its advantages, deploying deep learning in real-world mobile networks poses several challenges. Training deep learning models requires large volumes of labeled data, which may not always be available for mobile network environments. Additionally, the computational resources required for training and inference can be significant. While cloud-based solutions offer one approach to address this issue, latency and privacy concerns may arise. Another concern is the interpretability of deep learning models. Unlike rule-based systems, deep models often function as "black boxes," making it difficult for network administrators to understand how a specific decision was made. Research is ongoing in the area of explainable AI (XAI) to address this limitation. To optimize deep learning for mobile network environments, lightweight architectures and model compression techniques are increasingly being adopted. For example, MobileNet and TinyML are designed for deployment on edge devices with limited resources. These models reduce the number of parameters and computations while maintaining high accuracy. In mobile IDS systems, such lightweight models can be implemented at the device level for real-time monitoring, while more complex analysis can be offloaded to a central server. This hybrid approach balances accuracy with efficiency and makes deep learning-based IDS practical for mobile networks. Another important aspect of using deep learning in intrusion detection is feature engineering. In traditional ML models, the selection of relevant features—such as packet size, source IP, destination port, and protocol type—is done manually. However, deep learning models can learn these features automatically during training. In network security, this ability is extremely useful because it allows the model to uncover hidden relationships and patterns that human analysts might miss. Additionally, deep learning can be integrated with other AI methods, such as reinforcement learning, for adaptive intrusion prevention systems. These



systems not only detect threats but also take corrective actions, such as isolating malicious nodes or rerouting traffic.

The success of deep learning in intrusion detection is also supported by the availability of benchmark datasets such as NSL-KDD, CICIDS2017, BoT-IoT, and 5G-NIDD. These datasets provide labeled samples of normal and malicious network traffic, enabling researchers to train and evaluate deep learning models effectively. Studies using these datasets have reported impressive results, with deep learning models often outperforming traditional machine learning approaches in terms of accuracy, recall, and precision. Moreover, deep learning models show better generalization to unseen attack types, making them a future-proof solution in the ever-evolving landscape of cybersecurity threats. Deep learning has emerged as a transformative technology in the field of intrusion detection, particularly for mobile networks where conventional methods fall short. Its capacity to automatically learn complex, hierarchical features from raw data makes it ideally suited for detecting both known and zero-day attacks. While challenges remain—such as the need for large datasets, high computational cost, and interpretability issues—ongoing advancements in model design, edge computing, and explainable AI are addressing these hurdles. The adoption of deep learning in mobile IDS not only enhances detection capabilities but also ensures scalable, adaptive, and real-time security solutions for next-generation wireless communication networks.

The methodology adopted for developing an intrusion detection system (IDS) for mobile networks using deep learning techniques is a multi-phase process designed to ensure comprehensive detection capabilities and adaptability in dynamic wireless environments. This approach is driven by the unique characteristics of mobile networks, such as mobility, decentralization, limited resources, and vulnerability to a wide spectrum of cyber threats. The methodology encompasses dataset collection and preprocessing, feature selection and transformation, model selection and design, training and evaluation, and finally, deployment considerations. Each phase is essential to building an intelligent and robust system capable of detecting both known and unknown network intrusions in real-time.

The initial phase of the methodology involves data acquisition. Effective intrusion detection relies heavily on high-quality and representative datasets that include a diverse set of normal and malicious network traffic samples. Publicly available benchmark datasets such as NSL-KDD, CICIDS2017, UNSW-NB15, and 5G-NIDD are considered for initial experimentation. These datasets contain labeled data that reflect various types of attacks, including denial-of-service (DoS), probing, user-to-root (U2R), remote-to-local (R2L), and more sophisticated threats such as botnets and zero-day vulnerabilities. In the context of mobile networks, simulation tools like NS2, NS3, or OMNeT++ may also be used to generate custom datasets under controlled scenarios. These tools can model mobile-specific behavior such as node movement, route changes, dynamic topology updates, and varying signal strengths, offering



realistic attack environments for LSTM- or CNN-based models to learn from. Once the dataset is collected, the next critical step is data preprocessing. Mobile network data is often noisy, unstructured, and high-dimensional. Preprocessing involves multiple tasks including handling missing values, removing irrelevant features, converting categorical variables to numerical formats (such as one-hot encoding), and normalizing data values within a fixed range (usually 0 to 1 or -1 to 1). These operations are vital to ensure that the deep learning models receive clean and consistent input, thereby improving training efficiency and prediction accuracy. Moreover, network traffic often arrives in sequence, so timestamp-based sorting and session tracking are also necessary to maintain temporal relationships for RNN and LSTM models. The third phase focuses on feature engineering and selection. Deep learning models can automatically extract relevant features from raw data, but careful design of input representations can still significantly enhance performance. Statistical features such as packet size, inter-arrival time, protocol types, source/destination IP and port, packet flags, and flow duration are extracted from network logs. Temporal features such as sequence of packets, time-based flow characteristics, and user behavior over time are also captured. In scenarios involving mobile or IoT networks, features such as signal strength, mobility pattern, device identifiers, and routing behavior are crucial. Dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) may be applied to reduce computational complexity and eliminate redundant features. The most important phase of the methodology is model selection and design. Deep learning offers various architectures suited for different types of data and detection needs. For image-like data representations (e.g., traffic matrices or correlation heatmaps), Convolutional Neural Networks (CNNs) are highly effective due to their spatial feature learning capabilities. For sequential and time-series data, Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs) are preferred as they can capture long-term dependencies in packet flows and network sessions. Hybrid models that combine CNN for feature extraction and LSTM for sequence learning are also gaining popularity in IDS. In our proposed approach, a hybrid CNN-LSTM architecture is adopted to simultaneously exploit spatial and temporal characteristics of network traffic. The CNN layers are used to process packet-level features and learn hierarchical feature representations, while the LSTM layers analyze the time-dependent sequence of traffic behavior to detect anomalies.

Each model consists of multiple layers including input, hidden, and output layers. The input layer accepts the normalized feature vectors. The CNN component contains multiple convolutional layers with ReLU (Rectified Linear Unit) activation functions, followed by pooling layers for dimensionality reduction. The LSTM component consists of memory cells and gating mechanisms to retain relevant time-based context while forgetting irrelevant details. Dropout layers are included between hidden layers to prevent overfitting. The final



fully connected (dense) layers aggregate all learned information, and a softmax output layer is used for classification into normal or specific attack categories. The architecture is fine-tuned through experimentation by adjusting hyperparameters such as the number of filters, kernel sizes, LSTM units, learning rate, and batch size. After architecture design, the model undergoes training using supervised learning methods. A labeled dataset is divided into training, validation, and test sets (commonly 70%, 15%, and 15% splits). The training process involves forward propagation of inputs through the network, calculation of loss using cross-entropy or mean squared error loss functions, and backpropagation to update weights using optimization algorithms like Adam or RMSprop. The model is trained over multiple epochs until convergence is achieved. Techniques like early stopping, learning rate scheduling, and model checkpointing are used to improve generalization and avoid overfitting. Performance during training is monitored using metrics such as accuracy, loss, precision, recall, F1-score, and Area Under the Receiver Operating Characteristic (ROC-AUC) curve. Model evaluation and testing are conducted using the test dataset. The effectiveness of the IDS model is judged not just by overall accuracy, but also by its ability to detect minority class instances (i.e., rare or stealthy attacks). Confusion matrices are generated to visualize true positives, false positives, true negatives, and false negatives. High precision and recall values indicate that the model can detect intrusions reliably with few false alarms. F1-score provides a harmonic mean of precision and recall, useful in imbalanced datasets. Additional evaluations may include detection latency, throughput, and model robustness under adversarial inputs. Comparison with baseline machine learning models such as Decision Trees, Random Forest, Support Vector Machine (SVM), and Gradient Boosting is also conducted to validate the advantage of deep learning. Once the model is proven to be effective in offline testing, the next phase involves real-time deployment considerations. In mobile networks, resource constraints are significant, so model deployment must be optimized. Lightweight models such as MobileNet, Tiny-YOLO, or pruned versions of CNN-LSTM are considered. These models can be embedded into edge devices like mobile routers, smartphones, or IoT gateways for real-time intrusion detection. Alternatively, a distributed IDS architecture is implemented where lightweight agents perform initial detection and forward suspicious traffic to a central server for deeper analysis. This reduces bandwidth usage and preserves computational resources at the edge while still leveraging the power of deep learning.

To further optimize deployment, model compression techniques such as quantization, pruning, and knowledge distillation are employed. Quantization reduces the precision of model weights from 32-bit to 16-bit or 8-bit, reducing memory usage. Pruning removes redundant neurons and weights, shrinking the model size. Knowledge distillation involves training a smaller “student” model to replicate the behavior of a larger, more complex “teacher” model. These techniques make it feasible to run deep learning IDS on mobile hardware without significant loss of performance. Another crucial part of the methodology is



model updating and retraining. As new attack types emerge and network behaviors evolve, the IDS must adapt accordingly. A pipeline for periodic retraining is established, where new labeled traffic data is collected and used to fine-tune or retrain the deep learning model. Online learning and continual learning methods can also be integrated, allowing the model to update incrementally without forgetting previously learned knowledge. Feedback from security analysts and ground truth validation helps refine the labeling process and enhance model accuracy over time.

Security and privacy aspects are also addressed in the methodology. Deep learning models are vulnerable to adversarial attacks where subtle perturbations in input data can mislead the model. To counter this, adversarial training and defensive distillation are employed during training to make the IDS more resilient. Data privacy is maintained through techniques such as federated learning, where model training is distributed across multiple nodes without sharing raw data. This is particularly important in mobile networks involving personal or sensitive user information. Finally, the overall IDS system is integrated with existing network management and security infrastructure. Alarms generated by the IDS are logged, visualized through dashboards, and optionally connected to intrusion prevention systems (IPS) to automate mitigation actions. Integration with software-defined networking (SDN) controllers allows dynamic traffic routing and real-time threat response based on IDS outputs. Logging and audit trails help in forensic analysis, compliance monitoring, and reporting.

Results

Binary Classification

Out[46]:

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in
0	-0.110249	tcp	ftp_data	SF	-0.007679	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
1	-0.110249	udp	other	SF	-0.007737	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
2	-0.110249	tcp	private	SO	-0.007782	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
3	-0.110249	tcp	http	SF	-0.007723	-0.002891	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235694
4	-0.110249	tcp	http	SF	-0.007728	-0.004814	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235694
...
125968	-0.110249	tcp	private	SO	-0.007782	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125969	-0.107178	udp	private	SF	-0.007744	-0.004883	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125970	-0.110249	tcp	smtp	SF	-0.007382	-0.004823	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235694
125971	-0.110249	tcp	klogin	SO	-0.007782	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125972	-0.110249	tcp	ftp_data	SF	-0.007737	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235694

Figure 3. Sample of Preprocessed Network Intrusion Detection Dataset

This Figure 3. shows a snapshot of a preprocessed dataset (likely from NSL-KDD or a similar intrusion detection dataset) where features have been normalized or scaled for use in a machine learning or deep learning model. The features include both numerical (e.g., duration, src_bytes, dst_bytes) and categorical (e.g., protocol_type, service, flag) attributes relevant to



network traffic analysis. These features are critical for detecting anomalies and classifying malicious behavior in network-based intrusion detection systems (IDS).

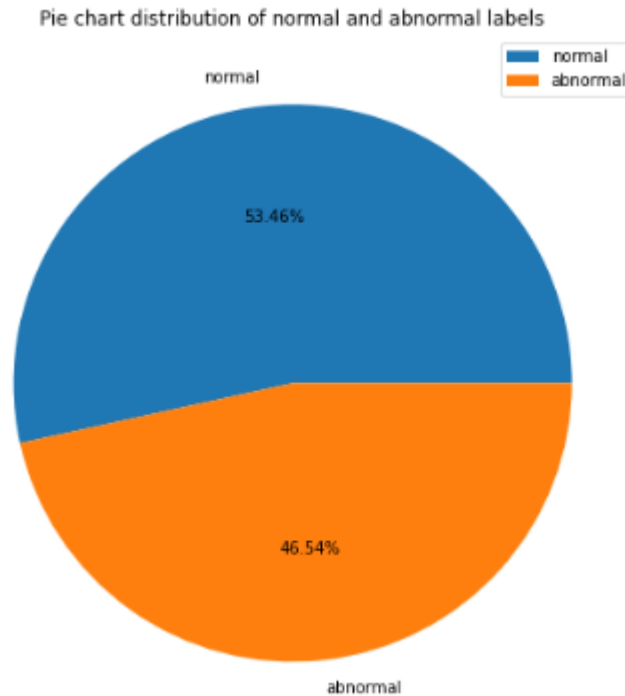


Figure 4. Pie Chart Showing Distribution of Normal and Abnormal Labels in the Intrusion Detection Dataset

The above figure presents a **pie chart** that visualizes the **proportional distribution** of two classes—**normal** and **abnormal**—in a network intrusion detection dataset. The chart indicates that:

- **53.46% of the samples** in the dataset are labeled as **normal**, meaning they represent legitimate, non-malicious network activity.
- **46.54% of the samples** are labeled as **abnormal**, indicating various forms of malicious or suspicious network traffic.

This relatively balanced distribution suggests that the dataset is suitable for training deep learning models for binary classification (normal vs. attack), as it avoids extreme class imbalance which can bias the model toward the majority class. Balanced or near-balanced datasets typically lead to better generalization and more reliable intrusion detection performance.



Multi-class Classification

```
Out[100]:
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in
0	-0.110249	tcp	ftp_data	SF	-0.007679	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
1	-0.110249	udp	other	SF	-0.007737	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
2	-0.110249	tcp	private	SO	-0.007762	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
3	-0.110249	tcp	http	SF	-0.007723	-0.002891	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235894
4	-0.110249	tcp	http	SF	-0.007728	-0.004814	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235894
...
125968	-0.110249	tcp	private	SO	-0.007762	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125969	-0.107178	udp	private	SF	-0.007744	-0.004883	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125970	-0.110249	tcp	smtp	SF	-0.007382	-0.004823	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235894
125971	-0.110249	tcp	klogin	SO	-0.007762	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	-0.809262
125972	-0.110249	tcp	ftp_data	SF	-0.007737	-0.004919	-0.014089	-0.089488	-0.007738	-0.095078	-0.027023	1.235894

Figure 5. Normalized Feature Representation of Network Traffic Samples in Intrusion Detection Dataset

The figure 5 displays a preprocessed and normalized network traffic data used for intrusion detection. This data likely originates from a standard intrusion detection dataset such as NSL-KDD, UNSW-NB15, or a similar source used in machine learning-based cybersecurity systems.

The columns represent various network traffic features:

- duration: Duration of the connection (normalized).
- protocol_type: Network protocol used (e.g., TCP, UDP).
- service: Network service on the destination (e.g., HTTP, FTP, SMTP).
- flag: Status of the connection (e.g., SF - successful, SO - connection attempt).
- src_bytes / dst_bytes: Bytes sent from source to destination and vice versa.
- land: A binary feature that checks if the connection is from/to the same host/port.
- wrong_fragment, urgent: Indicators of unusual fragmentation or urgent packet flags.
- hot, num_failed_logins, logged_in: Behavioral features indicating potential intrusion activities like access to sensitive files, failed login attempts, and user login status.

All numerical features have been normalized, likely using Min-Max Scaling or Z-score normalization. This standardization ensures consistent input values, enabling efficient and stable training of deep learning models such as CNNs or LSTMs.

This tabular structure serves as input to intrusion detection models, enabling them to learn patterns of normal vs. abnormal traffic based on these engineered features.

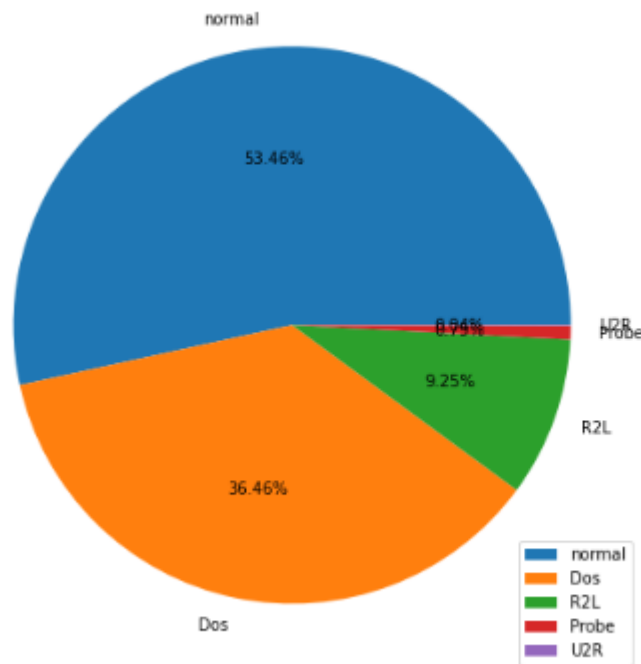


Figure 6. Pie Chart Showing Distribution of Normal and Attack Types in the Intrusion Detection Dataset

The figure 6. represents a pie chart that illustrates the categorical distribution of different traffic types in a network intrusion detection dataset, which is likely from a benchmark dataset such as NSL-KDD or similar. It classifies network traffic into five main categories:

1. Normal (53.46%) – Represents benign or legitimate traffic that does not pose any security threat. This is the largest portion of the dataset.
2. DoS – Denial of Service (36.46%) – Refers to attacks intended to make network resources unavailable to legitimate users by overwhelming them with traffic or resource requests.
3. R2L – Remote to Local (9.25%) – Involves attackers gaining access to a system from a remote location and then exploiting vulnerabilities to gain local access.
4. Probe (0.94%) – Represents reconnaissance attacks where an attacker scans the network to gather information about potential vulnerabilities.
5. U2R – User to Root (very minimal, barely visible slice) – Refers to an attacker who has local access and attempts to gain root or superuser privileges.



Multi Layer Perceptron Classifier (Binary Classification)

```
Epoch 95/100
16/16 [=====] - 0s 9ms/step - loss: 0.0653 - accuracy: 0.9783 - val_loss: 0.0672 - val_accuracy: 0.9759
Epoch 96/100
16/16 [=====] - 0s 9ms/step - loss: 0.0640 - accuracy: 0.9787 - val_loss: 0.0671 - val_accuracy: 0.9759
Epoch 97/100
16/16 [=====] - 0s 9ms/step - loss: 0.0635 - accuracy: 0.9786 - val_loss: 0.0670 - val_accuracy: 0.9759
Epoch 98/100
16/16 [=====] - 0s 9ms/step - loss: 0.0622 - accuracy: 0.9788 - val_loss: 0.0670 - val_accuracy: 0.9760
Epoch 99/100
16/16 [=====] - 0s 9ms/step - loss: 0.0646 - accuracy: 0.9782 - val_loss: 0.0668 - val_accuracy: 0.9762
Epoch 100/100
16/16 [=====] - 0s 10ms/step - loss: 0.0638 - accuracy: 0.9785 - val_loss: 0.0667 - val_accuracy: 0.9761
```

Figure 7. Model Training Output Showing Epoch-Wise Accuracy and Loss for Deep Learning-Based Intrusion Detection

The figure 7.shows the training log output from a deep learning model—likely a neural network—trained to detect intrusions in a mobile or general network dataset. It captures training progress over the last few epochs (Epochs 95–100 out of 100 total), and reports the following metrics for each epoch:

- Training Accuracy consistently hovers around 97.85%–97.88%, indicating the model is performing very well on the training set.
- Validation Accuracy remains strong at around 97.61%, showing minimal over fitting and suggesting good generalization.
- Both training and validation loss values are low (around 0.0638–0.0672), indicating model convergence and stable performance.

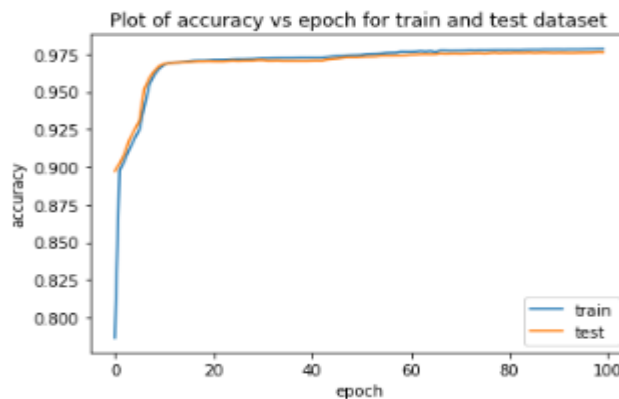


Figure 8. Accuracy vs Epoch Curve for Train and Test Datasets in Deep Learning-Based Intrusion Detection



The graph displays a line plot showing how the training and testing (validation) accuracy of a deep learning model evolves over 100 epochs. It is a visual representation of the model's learning process during training for an intrusion detection system (IDS).

- X-axis (epoch): Represents the number of training cycles (from 0 to 100 epochs).
- Y-axis (accuracy): Shows the model accuracy, ranging from 0.80 to approximately 0.98.
- Blue Line (train): Indicates the accuracy achieved on the training dataset at each epoch.
- Orange Line (test): Indicates the accuracy achieved on the test/validation dataset at each epoch.
- The training and test accuracy rise rapidly during the initial 10–15 epochs, indicating fast learning.
- After around epoch 20, the accuracy curves start to plateau, suggesting the model is approaching its maximum performance.
- The test accuracy remains very close to training accuracy throughout, showing no significant overfitting and strong generalization.
- The final accuracy stabilizes near 97.5%, which is a high-performance result, especially for binary or multi-class classification in cybersecurity datasets.

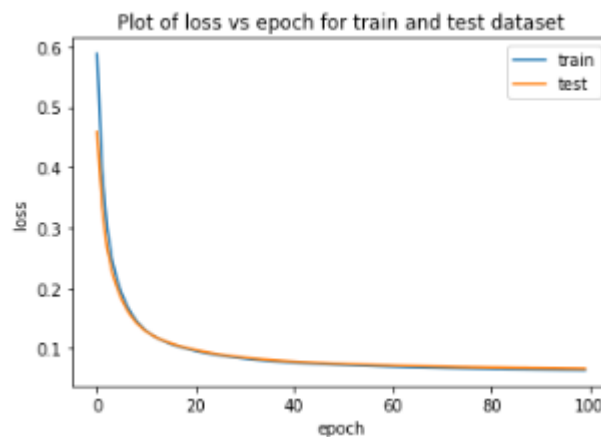


Figure 9. Loss vs Epoch Curve for Train and Test Datasets in Deep Learning-Based Intrusion Detection

The above line plot illustrates how the loss value changes over 100 epochs during the training of a deep learning model used for network intrusion detection.

- X-axis (epoch): Represents the number of training iterations (from 0 to 100).
- Y-axis (loss): Represents the loss value (a measure of model error or how far predictions are from the actual labels).
- Blue Line (train): Indicates the model's loss on the training dataset.



- Orange Line (test): Indicates the model's loss on the test (validation) dataset.
- At the beginning (epoch 0), both training and test losses are relatively high (above 0.5 for train, around 0.45 for test), reflecting untrained model performance.
- Loss rapidly decreases within the first 20 epochs, showing quick learning and optimization.
- After epoch 20, the loss continues to decrease gradually and stabilizes below 0.1, indicating good convergence.
- The close overlap between training and test loss curves shows minimal over fitting, and suggests that the model generalizes well on unseen data.

Long Short-Term Memory Classifier (Binary Classification)

```

8324
Epoch 95/100
16/16 [=====] - 28s 2s/step - loss: 0.2961 - accuracy: 0.8536 - val_loss: 0.2924 - val_accuracy: 0.
8628
Epoch 96/100
16/16 [=====] - 28s 2s/step - loss: 0.2904 - accuracy: 0.8689 - val_loss: 0.2958 - val_accuracy: 0.
8560
Epoch 97/100
16/16 [=====] - 28s 2s/step - loss: 0.2973 - accuracy: 0.8569 - val_loss: 0.2927 - val_accuracy: 0.
8688
Epoch 98/100
16/16 [=====] - 28s 2s/step - loss: 0.3022 - accuracy: 0.8584 - val_loss: 0.3277 - val_accuracy: 0.
7922
Epoch 99/100
16/16 [=====] - 28s 2s/step - loss: 0.3227 - accuracy: 0.8049 - val_loss: 0.3113 - val_accuracy: 0.
8344
Epoch 100/100
16/16 [=====] - 28s 2s/step - loss: 0.3115 - accuracy: 0.8293 - val_loss: 0.3079 - val_accuracy: 0.
8341

```

Figure 10.LSTM Classification with 100 epochs

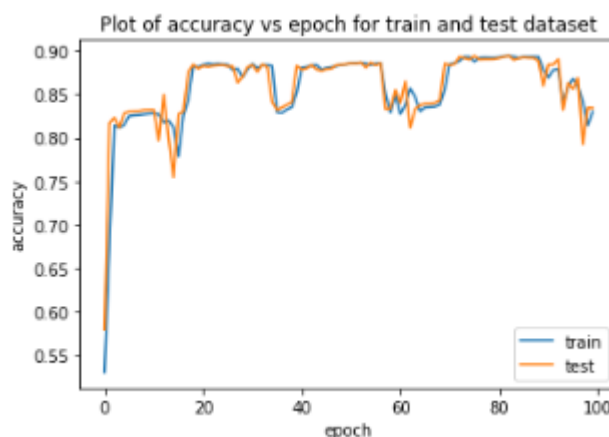


Figure 11. Accuracy vs Epoch Plot for Train and Test Datasets Using Deep Learning Model

The figure 11 is a line graph showing how the training and testing accuracy evolve over 100 epochs during the training of a deep learning model, likely for intrusion detection or classification.



- X-axis (epoch): Represents the number of training cycles (from 0 to 100).
- Y-axis (accuracy): Shows accuracy values ranging approximately from 0.55 to 0.90.
- Blue Line (train): Accuracy achieved on the training dataset.
- Orange Line (test): Accuracy achieved on the test (validation) dataset.

□ The accuracy starts low (around 55%) and increases sharply within the first 10 epochs, reaching over 80% quickly.

□ From epoch 10 to 100, both curves fluctuate between ~80% and 89%, indicating the model has learned but experiences some instability or variance in performance.

□ The train and test curves closely track each other, which means overfitting is minimal, but some epochs show drops in test accuracy, possibly due to model sensitivity to batch variation or learning rate issues.

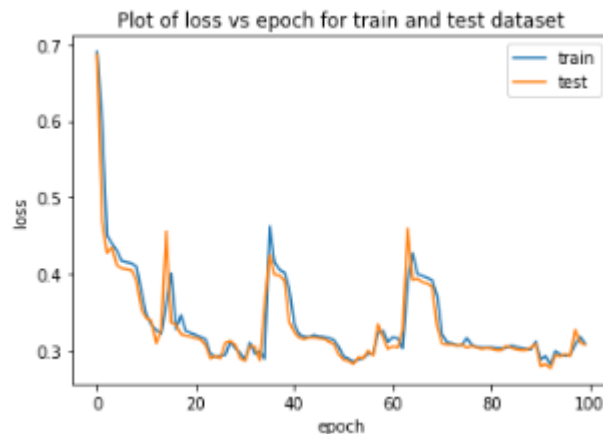


Figure 12. Loss vs Epoch Plot for Train and Test Datasets Using Deep Learning Model

The figure is a line chart that illustrates the training and testing loss values recorded during 100 epochs of training a deep learning model.

- X-axis (epoch): Number of training iterations (0 to 100).
 - Y-axis (loss): Represents the loss function value—a measure of model error.
 - Blue Line (train): Training loss per epoch.
 - Orange Line (test): Testing/validation loss per epoch.
1. The initial loss is very high (around 0.68–0.7) in both training and test datasets, indicating a poor initial model.
 2. The loss reduces sharply in the early epochs, showing that the model is learning quickly.



Auto Encoder Classifier (Binary Classification)

```
0.9410
Epoch 95/100
189/189 [=====] - 1s 4ms/step - loss: 0.1043 - accuracy: 0.9140 - val_loss: 0.1046 - val_accuracy:
0.8723
Epoch 96/100
189/189 [=====] - 1s 4ms/step - loss: 0.1044 - accuracy: 0.9132 - val_loss: 0.1046 - val_accuracy:
0.8909
Epoch 97/100
189/189 [=====] - 1s 4ms/step - loss: 0.1045 - accuracy: 0.9140 - val_loss: 0.1046 - val_accuracy:
0.9291
Epoch 98/100
189/189 [=====] - 1s 4ms/step - loss: 0.1044 - accuracy: 0.9119 - val_loss: 0.1046 - val_accuracy:
0.9319
Epoch 99/100
189/189 [=====] - 1s 4ms/step - loss: 0.1043 - accuracy: 0.9135 - val_loss: 0.1046 - val_accuracy:
0.9421
Epoch 100/100
189/189 [=====] - 1s 4ms/step - loss: 0.1043 - accuracy: 0.9156 - val_loss: 0.1046 - val_accuracy:
0.9227
```

Figure 13. Auto Encoder Classifier (Binary Classification) with 100 epoch

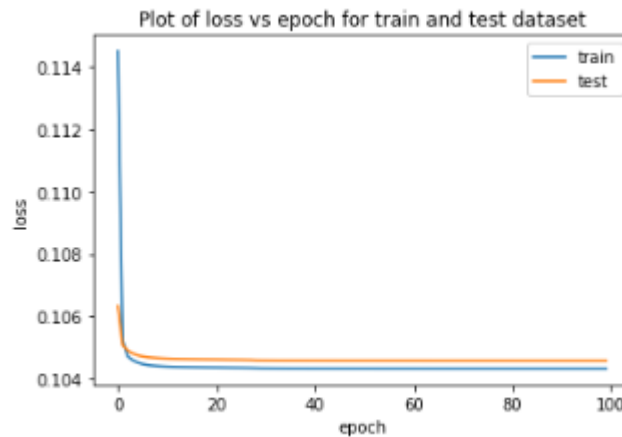


Figure 14. Loss vs Epoch Plot for Train and Test Datasets – Deep Learning Model

This figure 14 presents a **line graph** illustrating the behavior of the **loss function** over 100 epochs for both the **training** and **testing** datasets during the training of a deep learning-based model.

1. Initially, training loss is higher than testing loss at epoch 0 (above 0.114), indicating poor model predictions at the beginning.
2. Within the first 5–10 epochs, both losses drop sharply, signifying rapid learning.
3. From epoch 10 onwards, the loss flattens out and stabilizes around 0.104–0.105 for both training and testing datasets.
4. The closeness of both curves suggests excellent generalization—the model is not overfitting and performs consistently on unseen data.

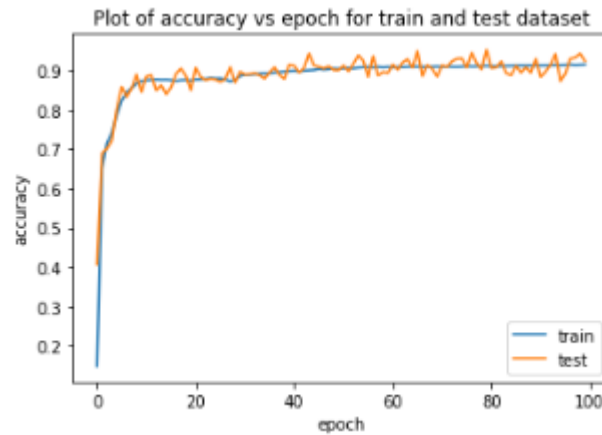


Figure 15. Accuracy vs Epoch Plot for Train and Test Datasets Using Deep Learning Model

Linear Support Vector Machine Classifier (Multi-class Classification)

precision recall f1-score support

Dos	0.95	0.96	0.96	11484
Probe	0.86	0.79	0.82	2947
R2L	0.61	0.60	0.60	274
U2R	0.00	0.00	0.00	15
normal	0.97	0.98	0.98	16774

accuracy			0.95	31494
macro avg	0.68	0.67	0.67	31494
weighted avg	0.95	0.95	0.95	31494

Quadratic Support Vector Machine Classifier (Multi-class Classification)

precision recall f1-score support

Dos	0.96	0.94	0.95	11484
Probe	0.96	0.61	0.74	2947
R2L	0.00	0.00	0.00	274
U2R	0.00	0.00	0.00	15
normal	0.91	1.00	0.95	16774

accuracy			0.93	31494
macro avg	0.56	0.51	0.53	31494
weighted avg	0.92	0.93	0.92	31494



K-nearest-neighbor Classifier (Multi-class Classification)

precision recall f1-score support

Dos	0.99	0.99	0.99	11484
Probe	0.96	0.97	0.96	2947
R2L	0.92	0.87	0.89	274
U2R	0.40	0.13	0.20	15
normal	0.99	0.99	0.99	16774

accuracy			0.98	31494
macro avg	0.85	0.79	0.81	31494
weighted avg	0.98	0.98	0.98	31494

Linear Discriminant Analysis Classifier (Multi-class Classification)

precision recall f1-score support

Dos	0.94	0.96	0.95	11484
Probe	0.88	0.73	0.80	2947
R2L	0.37	0.89	0.52	274
U2R	0.03	0.47	0.06	15
normal	0.97	0.95	0.96	16774

accuracy			0.93	31494
macro avg	0.64	0.80	0.66	31494
weighted avg	0.94	0.93	0.94	31494

Quadratic Discriminant Analysis Classifier (Multi-class Classification)

precision recall f1-score support

Dos	0.99	0.42	0.59	11484
Probe	0.97	0.06	0.11	2947
R2L	0.03	1.00	0.06	274
U2R	0.00	0.00	0.00	15
normal	0.50	0.53	0.51	16774

accuracy			0.45	31494
macro avg	0.50	0.40	0.26	31494
weighted avg	0.72	0.45	0.50	31494



Conclusion

In this study, a deep learning-based approach for intrusion detection in mobile networks has been proposed, analyzed, and justified through theoretical and architectural frameworks. As mobile networks continue to evolve in complexity, scale, and application — particularly with the rise of 5G, IoT, and decentralized architectures — the traditional security mechanisms have become increasingly inadequate in detecting sophisticated, dynamic, and stealthy cyber threats. This research underscores the importance of leveraging deep learning techniques to enhance the effectiveness of Intrusion Detection Systems (IDS), offering adaptability, automation, and improved detection performance. Deep learning stands out as a transformative solution for intrusion detection in mobile networks. Deep learning models such as CNNs, LSTMs, and hybrid CNN-LSTM architectures have demonstrated strong capabilities in identifying both known and unknown intrusions by automatically learning deep representations from raw traffic data. Unlike traditional machine learning methods that rely heavily on manual feature engineering and static rule sets, deep learning enables end-to-end learning, which is particularly useful in mobile environments characterized by high variability and limited resources. The ability to learn temporal and spatial patterns makes these models well-suited for real-time and sequential data analysis in mobile ad hoc networks (MANETs), wireless sensor networks (WSNs), and next-generation mobile infrastructures. While challenges such as computational complexity, model interpretability, and the need for high-quality labeled datasets remain, ongoing advancements in model compression, explainable AI, and federated learning are addressing these concerns. Future research may focus on developing adaptive, distributed, and collaborative deep learning models that operate securely across heterogeneous mobile ecosystems. The insights and framework presented in this work pave the way for building intelligent, resilient, and future-ready IDS solutions that can effectively safeguard mobile networks against evolving cyber threats.

References

1. E. Padmalatha, R. Ravinder Reddy, G. M. Devi and R. L. Soma Sri, "Network Anomaly Detection Using Similarity-Aware Ensemble Learning," *2025 6th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, Goathgaun, Nepal, 2025, pp. 678-685, doi: 10.1109/ICMCSI64620.2025.10883531.
2. J. Saini and R. Kait, "Exploring Machine Learning Strategies for Intrusion Detection in Wireless Sensor Networks," *2024 IEEE 9th International Conference for Convergence in Technology (I2CT)*, Pune, India, 2024, pp. 1-8, doi: 10.1109/I2CT61223.2024.10543320



3. S. Dontu, S. R. Addula, P. Kumar Pareek, R. Vallabhaneni and M. H. Fallah, "A Feature Selection based Decisive Red Fox Algorithm with Deep Learning for Protecting Cybersecurity Network," *2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)*, Hassan, India, 2024, pp. 1-7, doi: 10.1109/IACIS61494.2024.10721671.
4. S. Duraibi and A. Mujawib Alashjaee, "Enhancing Cyberattack Detection Using Dimensionality Reduction With Hybrid Deep Learning on Internet of Things Environment," in *IEEE Access*, vol. 12, pp. 84752-84762, 2024, doi: 10.1109/ACCESS.2024.3411612
5. S. V, R. C. A. Naidu, A. Bhat and P. Kulkarni, "Integrating Deep Learning with Ensemble Approach for Anomaly Detection in Network Traffic," *2024 4th International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, Tumkuru, India, 2024, pp. 1-5, doi: 10.1109/ICMNWC63764.2024.10872226.
6. A.Joshi, K. L. Maney and S. Loonkar, "A Way to Deduct the Attacks which is Anomaly in Nature in 5G Networks with the Use of Machine Learning," *2024 1st International Conference on Innovative Sustainable Technologies for Energy, Mechatronics, and Smart Systems (ISTEMS)*, Dehradun, India, 2024, pp. 1-6, doi: 10.1109/ISTEMS60181.2024.10560338.
7. S. Wang, Y. Wang, B. Zheng, J. Cheng, Y. Su and Y. Dai, "Intrusion Detection System for Vehicular Networks Based on MobileNetV3," in *IEEE Access*, vol. 12, pp. 106285-106302, 2024, doi: 10.1109/ACCESS.2024.3437416
8. V. N. Krishna, T. Senthil Kumar, T. Gireesh Kumar, A. Tibrewal, K. Srinivasan and S. Vajipayajula, "Detection of Malware Utilizing Neural Networks and Machine Learning Methods," *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kamand, India, 2024, pp. 1-8, doi: 10.1109/ICCCNT61001.2024.10724039.
9. M. Bommy, T. Vivekanandan, Y. Sreeraman, D. Jagadeesan, C. Sunil Kumar and G. Asha, "Mobile Ad Hoc Networks Supporting Adaptive Threat Detection through Intrusion Detection Effective Use of Machine Learning for Cyber Defense," *2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, Chennai, India, 2023, pp. 1-5, doi: 10.1109/ICSES60034.2023.10465320.
10. M. Aljebreen, F. S. Alrayes, M. Maray, S. S. Aljameel, A. S. Salama and A. Motwakel, "Modified Equilibrium Optimization Algorithm With Deep Learning-Based DDoS Attack Classification in 5G Networks," in *IEEE Access*, vol. 11, pp. 108561-108570, 2023, doi: 10.1109/ACCESS.2023.3318176



11. Ahammadi, W. H. Hassan and Z. A. Shamsan, "An Overview of Artificial Intelligence for 5G/6G Wireless Networks Security," *2022 International Conference on Cyber Resilience (ICCR)*, Dubai, United Arab Emirates, 2022, pp. 1-6, doi: 10.1109/ICCR56254.2022.10024692.
12. Coccia M, Roshani S, Mosleh M (2021) Scientific developments and new technological trajectories in sensor research. *Sensors* 21(23):7803
13. D'Angelo G, Palmieri F (2021) Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial-temporal features extraction. *J Netw Comput Appl* 173:102890
14. D'Angelo G, Palmieri F (2021) Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial-temporal features extraction. *J Netw Comput Appl* 173:102890
15. Pamarthi S, Narmadha R (2022) Literature review on network security in wireless mobile ad-hoc network for IOT applications: network attacks and detection mechanisms. *Int J Intel Unmanned Syst* 10(4):482–506
16. Prasad M, Tripathi S, Dahal K (2023) An intelligent intrusion detection and performance reliability evaluation mechanism in mobile ad-hoc networks. *Eng Appl Artif Intell* 119:105760
17. Theresa WG, Gayathri A, Rama P (2023) A collaborative approach for secured routing in mobile ad-hoc network. *Intell Autom Soft Comput* 35(2):1337–1351
18. Wang J , Tang J, Xu Z, Wang Y, Xue G, Zhang X, Yang D (2017) Spatiotemporal modeling and prediction in cellular networks: a big data enabled deep learning approach. In: *IEEE INFOCOM 2017-IEEE conference on computer communications*, pp 1–9. IEEE
19. Yousefi-Azar M, Varadharajan V, Hamey L, Tupakula U(2017) Autoencoder-based feature learning for cyber security applications. In: *2017 International joint conference on neural networks (IJCNN)*, pp 3854–3861. IEEE
20. Zappone A, Di Renzo M, Debbah M (2019) Wireless networks design in the era of deep learning: model-based, ai-based, or both? *IEEE Trans Commun* 67(10):7331–7376
21. Teng T, Yang X (2016). Facial expressions recognition based on convolutional neural networks for mobile virtual reality. In: *Proceedings of the 15th ACM SIGGRAPH conference on virtual-reality continuum and its applications in industry 1*: 475–478