applied sciences

MDPI

Article

Smart Attacks Learning Machine Advisor System for Protecting Smart Cities from Smart Threats

Hussein Ali *, Omar M. Elzeki * and Samir Elmougy

Faculty of Computers and Information, Mansoura University, Mansoura 35516, Egypt; mougy@mans.edu.eg
* Correspondence: huussein.aliabd@gmail.com (H.A.); omar_m_elzeki@mans.edu.eg (O.M.E.).

Abstract: The extensive use of Internet of Things (IoT) technology has recently enabled the development of smart cities. Smart cities operate in real-time to improve metropolitan areas' comfort and efficiency. Sensors in these IoT devices are immediately linked to enormous servers, creating smart city traffic flow. This flow is rapidly increasing and is creating new cybersecurity concerns. Malicious attackers increasingly target essential infrastructure such as electricity transmission and other vital infrastructures. Software-Defined Networking (SDN) is a resilient connectivity technology utilized to address security concerns more efficiently. The controller, which oversees the flows of each appropriate forwarding unit in the SDN architecture, is the most critical component. The controller's flow statistics are thought to provide relevant information for building an Intrusion Detection System (IDS). As a result, we propose a five-level classification approach based on SDN's flow statistics to develop a **S**mart **A**ttacks **L**earning **M**achine **A**dvisor (**SALMA**) system for detecting intrusions and for protecting smart cities from smart threats. We use the Extreme Learning Machine (ELM) technique at all levels. The proposed system was implemented on the NSL-KDD and KDDCUP99 benchmark datasets, and achieved 95% and 99.2%, respectively. As a result, our approach provides an effective method for detecting intrusions in SDNs.

Keywords: Software-Defined Networking (SDN); NSL-KDD; smart city; Intrusion Detection System (IDS); Extreme Learning Machine (ELM); Internet of Things (IoT)

Citation: Ali, H.; Elzeki, O.M.; Elmougy, S. Smart Attacks Learning Machine Advisor System for Protecting Smart Cities from Smart Threats. *Appl. Sci.* **2022**, *12*, 6473. https://doi.org/10.3390/app12136473

Academic Editor: Vicente Julian

Received: 14 May 2022
Accepted: 19 June 2022
Published: 25 June 2022

1. Introduction

According to studies from throughout the globe, cities are growing in size and population [1]. As a result of the scarcity of services and resources, such as medical, transportation, environment, and education, daily living in metropolitan areas will become even more difficult. The phrase "smart city" is used to adopt and to apply mobile computing systems across all of a city's components and levels via realistic data management networks [1]. To become more innovative, cities emphasize the use of technology for networked data management, such as the Internet of Things (IoT) [2], cloud computing, and big data. These data management systems enhance several elements of operations and organizations in the smart city, including traffic control, sustainable resource management, life quality, and infrastructure. Novel strategies for effective data management necessarily need to be developed in order to achieve the long-term sustainability of these services in metropolitan settings. The following components contribute to a smart city:

1. Smart Infrastructure, including city facilities with embedded smart technology, e.g., buildings, streets, energy, and water networks, smart grids, and sensors, etc.
2. Smart individuals, including strategies for motivating individuals to be more creative and receptive to new ideas.
3. Smart Mobility: Transportation networks with increased embedded systems for genuine management and surveillance.

4. Smart Services: Using technology and ICT to provide services throughout the city in education, safety, surveillance, health, and tourism, etc.
5. Smart Governance: The formation of smart governments in metropolitan areas, facilitated by technology service engagement, delivery, and participation.
6. Smart Economy: Using technological advancement to help companies grow, create jobs, and expand their communities.
7. Smart Environment: Using information and communication technologies, and innovation to safeguard and to manage resources (emission control, pollution monitoring sensors, systems for waste management, and recycling, etc.).
8. Smart Living: Urban development that increases sustainability and life quality.

These elements are inextricably linked; therefore, data collection and infrastructure facilities must be integrated into the city's physical Infrastructure. Governance is required to manage such subsystems and to achieve the goal of digitalization [3] and**Error! Reference source not found.** depicts a multilayer design for smart cities in general [4]. From top to bottom, this n-tier design must incorporate both physical and soft infrastructures, and the following layers [3].

- Layer (1) Natural Environment: This refers to all of the natural characteristics in the city's location (sea, rivers, forests, landscape, and lakes, etc.).
- Layer (2) Hard Infrastructure (non-ICT-based): All of the recognized urban characteristics are included in this layer as a result of human activity, and are required for city functioning (water-energy-waste, roads, buildings, bridges, and utilities, etc.)
- Layer (3) Hard Infrastructure (ICT-based): this refers to all smart gear that is used to provide SSC services (servers, supercomputers, networks, sensors, and data centers, etc.)
- Layer (4) Services: A plethora of intelligent city services, categorized according to worldwide urban key performance indicators and grouped into the six aspects of smart cities.
- Layer (5) Soft Infrastructure: the endpoint that consumes the services.

Physical Infrastructure is represented in Layers 1, 2, and 3, while the rest of the layers represent Soft Infrastructure in the Figure 1.

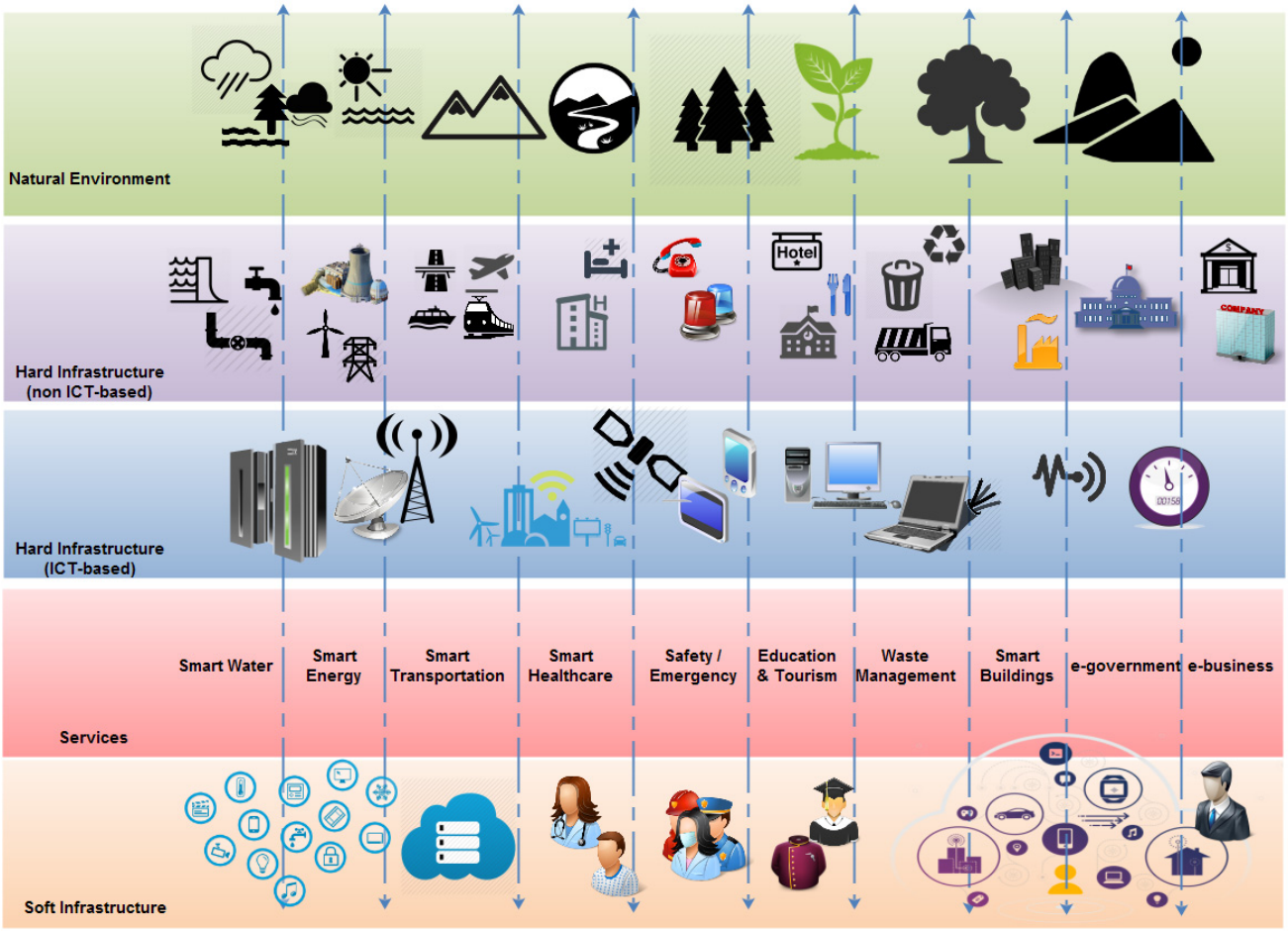

Figure 1. A smart city's generic multi-tiered ICT architecture [4].

Even as cities try to become smart, the aforementioned growing application technologies raise several security issues, and privacy problems and obstacles. Cyber-attacks in the smart city may affect any device linked to the network, which can have major issues when essential systems such as a city's electricity grid are connected. Confidentiality, integrity, availability, privacy, access control, and non-repudiation are the most crucial security and privacy concepts that need to be understood [5]. In the disciplines of information, communication, and physics, they must be met. Internet technology has advanced quickly in response to the advancement of new cyber-technologies. Hence, the age of interconnectedness with everything has been reached today. With the advent of Internet technology, we have entered a new generation of connectivity that has elevated connectivity to a critical and integral part of our modern life, giving ease and progressing civilization.

On the other hand, this technology has a slew of security issues brought on by hostile network attacks. According to Kaspersky Lab data released in the second quarter of 2018, over 962 million fraudulent intrusions were conducted in 187 nations worldwide, a steadily rising figure. Furthermore, cyber-attacks on mobile devices have demonstrated a random pattern of expansion because of the widespread use of mobile networking, exacerbating the severity of the crisis. Figure 2 shows that mobile threats in the second quarter of 2017 were around 1,300,000 threats, and this number kept increasing until it reached approximately 1,750,000 threats in the second quarter of 2018 [6], which illustrates how dealing with these intrusions by detecting and preventing them becomes more critical.

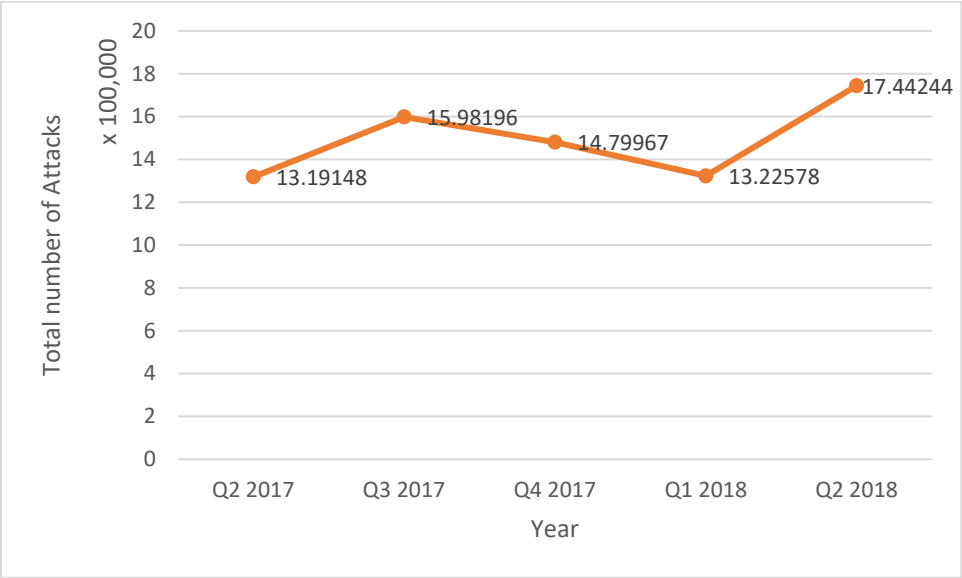

Figure 2. The situation of global mobile threats.

Globally, the adoption of IoT has recently risen dramatically. In 2017, the number of linked IoT devices topped 27 billion, and these IoT devices would continue to overgrow in response to market demand, potentially reaching roughly 125 billion by 2030 [7]. Malicious attackers increasingly target essential infrastructure such as smart cities, electricity transmission, and other vital infrastructures. There will be significant problems in the future if these facilities are vulnerable to cyber-attacks. As a result, protecting equipment against malware activity has become an essential and pressing task, as such intrusions may pose serious risks.

Intrusion Detection Systems (IDSs) have been a popular research area and a contentious problem, owing to the Internet's ever-increasing abundance of data. As a result, developing IDSs stands to reason, and it is a widely used operational security defensive strategy in the information industry. An IDS is a method/methodology for protecting application systems against malicious assaults, and it is the second line of defense. IDSs may be either network- or host-based. These may be used to defend a computer from the network or from end-user attacks. An IoT network and system, such as sensing equipment, might be an end-user device in the network [8]. Anomaly-based detection and signature-based detection are two types of detection techniques. By analyzing network traffic or data in computer memory for specified patterns, signature-based Detection approaches effectively recognize detected assaults. Anomaly-based detection monitors the behaviors of the whole objects, systems, or traffic, and compares them to predetermined behaviors that are thought to be normal, to find unknown threats. Any deviation from the usual operations is seen as a potential assault. Due to the complexity of attackers' techniques and the increase in zero-day assaults, an anomaly-based IDS is ideally suited to today's environment [9]. To detect abnormalities, most anomaly-based IDSs use Artificial Intelligence (AI), such as Machine Learning (ML) [9]. AI and Machine Learning aim to develop a computer that is capable of learning independently and distinguishing between normal and aberrant system behavior [10]. Many techniques for improving IDS performance have been advocated by academics, including Decision Tree (DT), artificial immune system, data mining, clustering-based methods, and statistics, etc.

Various applications link massive IoT devices to real-world items in smart cities. The huge number of IoT devices distributed over a broad array of protocols, devices, technologies, and services challenges the administration of the emerging IoT environment. As a result of these Internet integration protocols, substantial cybersecurity dangers and vulnerabilities exist for exploiting information regarding citizens' everyday activities [11]. At least two major security issues confront each smart application. The first challenge is in

recognizing zero-day attacks that start in a smart city's cloud data center via a range of protocols utilized by IoT devices, presuming that the massive attacks are hidden within IoT devices. The second difficulty is in identifying cyber-attacks intelligently (e.g., IoT malware attacks, etc.) [11] before IoT networks harm a smart city. Currently, most classic IDSs are still in use [12] because IoT network devices in smart cities have limited resources and capabilities, and are not suited for them [12].

However, updated hacker technology and sophisticated attack skills may create significant amounts of data with various features, including many samples, a variety of new attack kinds, and an unequal distribution of data. These issues are common in today's cyber environment, and they surely lower IDS performance [13]. ML can process multi-dimensional data and provide real-time predictions in dynamic contexts. Machine Learning developments have now broadened their application to include the creation of successful IDSs. Learning-based systems such as neural networks (NNs) have outperformed conventional approaches in various areas. Because of its learning and adaptable nature, ML-based IDSs can keep up with multiple threats.

This article describes an approach for detecting general-purpose network incursions using an Extreme Learning Machine (ELM) network. The bulk of previously announced intrusion detection algorithms expose only one form of attack at a time, or can separate legitimate traffic from attacks. A five-level classification model based on ELM algorithms is used in the proposed system. R2L and DOS ARE IDENTIFIED FIRST because their assaults have incredible detection accuracy. Then, since their models have lesser accuracies, both Probe and U2R assaults are in the following tiers. The suggested model is tested on the NSL-KDD [14] and utilizes one classifier per layer, and it may outperform popular shallow ML-based IDSs. This article employs a multilayer technique based on ELM classification algorithms. The following are some of this paper's most significant contributions:

- Developing a multi-level hybrid system capable of detecting intrusions based on six-flow characteristics that a standard Software-Defined Networking (SDN) controller can readily collect.
- Grouping multi-level homogenous classifiers hierarchically based on Machine Learning.
- Evaluating the suggested system's efficacy using standard datasets (NSL-KDD, KDDCUP99) that contain a collection of special attacks not included in the training set.
- Gains in accuracy are up to 95% compared to well-known state-of-the-art supervised Machine Learning methods employing similar datasets and flow-based features.

The remainder of the paper is organized as follows: the second section presents recent related work. The third section provides background information on smart cities. Section 4 examines anomaly detection methods; Section 5 gives a potential solution for anomaly detection; Section 6 discusses experiments and findings, and Section 7 concludes with a discussion of future directions.

2. Related Work

There are several attempts to help enhance the performance of various IDS(s) by utilizing Machine Learning, deep learning, and hybrid learning. This section discusses some recent research studies that propose an anomaly-based IDS and that use the NSL-KDD benchmark dataset to evaluate their proposed systems. Later, in the study, we compare our approach with these studies and show that our policy enhances them and achieves better results on the NSL-KDD benchmark dataset.

Tang et al. [15] used the NSL-KDD Dataset to train a Deep Neural Network (DNN) for flow-based anomaly detection in a Software-Defined Networking (SDN) model for IDSs [14]. They constructed a fundamental DNN consisting of three hidden layers, an input layer, and an output layer. They employed just six characteristics from the dataset

provided as an input of six neurons, including protocol type, src bytes, count, dst bytes, and srv count. These features include fundamental and traffic-based features that may be readily accessed in SDN setups. The binary classifier is the model's output. The neurons in the hidden layers are 12, 6, and 3. The batch size is set to 10 and the epoch is set to 100 in the model start parameters. The results of the experimentations achieved an accuracy of 75.75%. Rawat et al. [16] used DNN for IDSs in SDNs, which determined the most appropriate hyperparameters and network configurations in the DNN, and achieved 75.9% accuracy.

Wang et al. [17] presented the Localized Evolving Semi-Supervised Learning Based Anomaly Detection Scheme, called LESLA. Offline and online training were combined using semi-supervised learning in contrasting pessimistic likelihood estimation (CPLE) to allow self-evolution during anomaly detection. Hence, the self-training applies a hard label to the unlabeled data, causing the model to worsen, and increasing misclassification. Unlabeled data are soft-labeled in CPLE-based models. The model was trained using 500 labeled data and 120,000 unlabeled data, semi-supervised. Using NSL-KDD, the suggested system's accuracy is 80.93% when all of the dataset's features are used, and 77.26% when just six features are used.

Dey et al. [18] applied ML techniques in SDN architecture to suggest a flow-based anomaly detection solution in the OpenFlow controller. Some feature selection approaches, including the Chi-squared test, Symmetric Uncertainty, CFS Subset Evaluator, Gain Ratio, and Info Gain, were used to treat the dataset to enhance the classifier performance. Their research is based on NSL-KDD, which has 41 characteristics. Except for the CFS Subset Evaluator, which extracted nine features, all feature selection techniques extracted 15 features. The reduced dataset is then put through a variety of classifiers (J48, Random Forest (RF), PART, Naïve Bayes (NB), Decision Tree (DT), RBFN, and Bayes Net), while aggregating the results using 10-fold cross-validation. With the Gain Ratio for feature selection and RF for classification, this work achieved an accuracy of 82%.

Latah and Toker [19] evaluated different measures of well-known anomaly-based intrusion detection techniques. They concentrated on supervised ML algorithms that employ the following classifiers: RF, DT, NB, ELM, Support Vector Machines (SVM), K Nearest-Neighbor (KNN), NN, Linear Discriminant Analysis (LDA), Bagging Trees, LogitBoost, RUSBoost, and AdaBoost. They used the well-known NSL-KDD benchmark dataset. By focusing on just six SDN features, they utilized Principal Component Analysis (PCA) to de-dimensionalize the data. The most significant results were obtained while using Decision Tree, which has an accuracy of 83.24% and an f-score of 89.38%.

Tang et al. [20] presented a method for detecting intrusions into SDNs using a Gated Recurrent Unit Recurrent Neural Network (GRU-RNN). The Anomaly Mitigator, the Anomaly Detector, and the Flow Collector are the main components of this method. The Flow Collector module is invoked when a message contains a packet or when a timer function is called. It collects all flow data, including IP addresses, ports of the source and destination, and protocol details. The Anomaly Detector module will obtain all of the aggregated features, load a GRU-RNN trained model, receive network statistics, and determine whether or not a flow is an anomaly, allowing the Anomaly Mitigator module to make flow choices. GRU-RNN comprises four layers: an output layer and three hidden layers. They just employed six basic capabilities that are readily available in SDN settings. The output dimension is two, since the model is a binary classifier (Normal vs. Anomaly). Correspondingly, six, four, and two neurons are found in the buried layers. The suggested technique was tested using the NSL-KDD dataset and was shown to be 89% accurate.

Latah and Toker [21] proposed a five-layer classification approach that takes six flow features as input, as shown in Figure 3. They used kNN, ELM, and hierarchical ELM (HELM) for the first, second, and subsequent levels. They only employed the six flow characteristics immediately accessible via the controller. Their multilayer approach improved the system's overall accuracy to 84.29%, based on the NSL-KDD dataset.

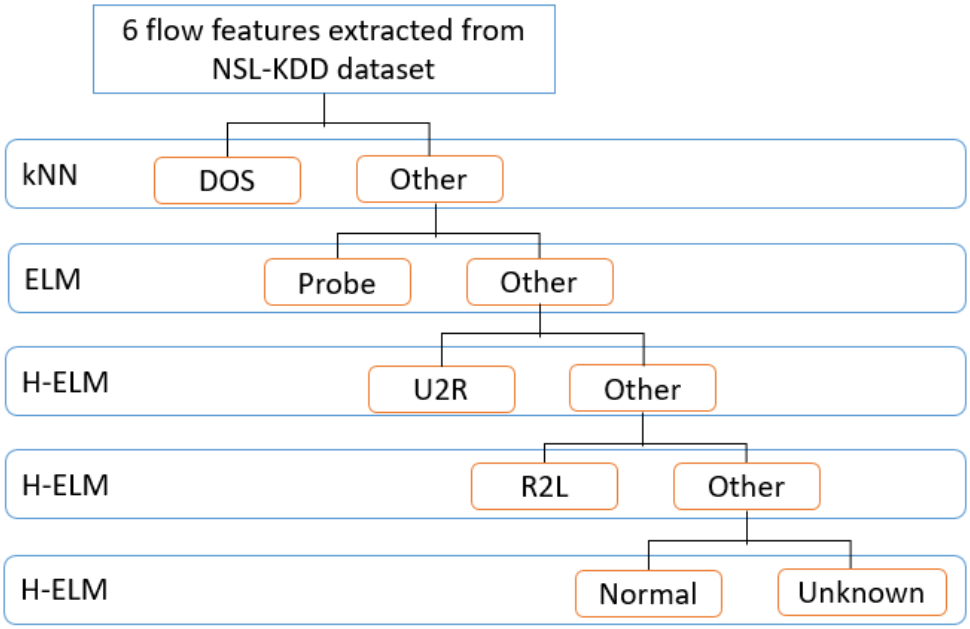

Figure 3. Multi-level hybrid IDS.

Gao et al. [6] presented a principal adaptive component (A-PCA) for an IDS-combined Incremental Extreme Learning Machine (I-ELM). The essential aspects of network traffic are adaptively picked in this technique, and the I-ELM then obtains the highest detection accuracy. Automatic feature extraction is accomplished using the A-PCA based on parameter limitations, while the I-ELM is in charge of identifying malicious assaults. The performance of this technique was evaluated using the NSL-KDD dataset. This work reached an accuracy of 81.22% and a detection rate of 96.1%.

Zheng et al. [22] proposed an improved LDA-based Extreme Learning Machine Classification (ILECA) as an IDS. They enhanced LDA by weighting it with a spatial similarity function to improve the between-class scatter matrix, then combined it with LDA to obtain the ideal transformation matrix for the spatial separation of high-dimensional data, and utilized it to minimize feature dimensions. They also identified the dimensionality-reduced data using ELM with the single hidden layer NN technique. They conducted experiments to validate their hypotheses using the NSL-KDD dataset. The evaluation findings indicated that the suggested ILECA had enhanced generalization characteristics, with an accuracy rate of 92.35%.

Al-Yaseen et al. [23] increased the effectiveness of identifying known and new threats, offering a multi-level hybrid intrusion detection model incorporating SVM and ELM (Figure 4). A modified K-means approach was also presented for creating a high-quality training dataset that considerably improves classifier performance. Modified K-means create new small training datasets that reflect the initial training dataset, reducing classifier training time and improving intrusion detection system performance. The proposed work's performance was evaluated using the KDDCUP99 benchmark dataset. Results showed that this work had an accuracy of 95.75% and a false alarm rate of 1.87%.

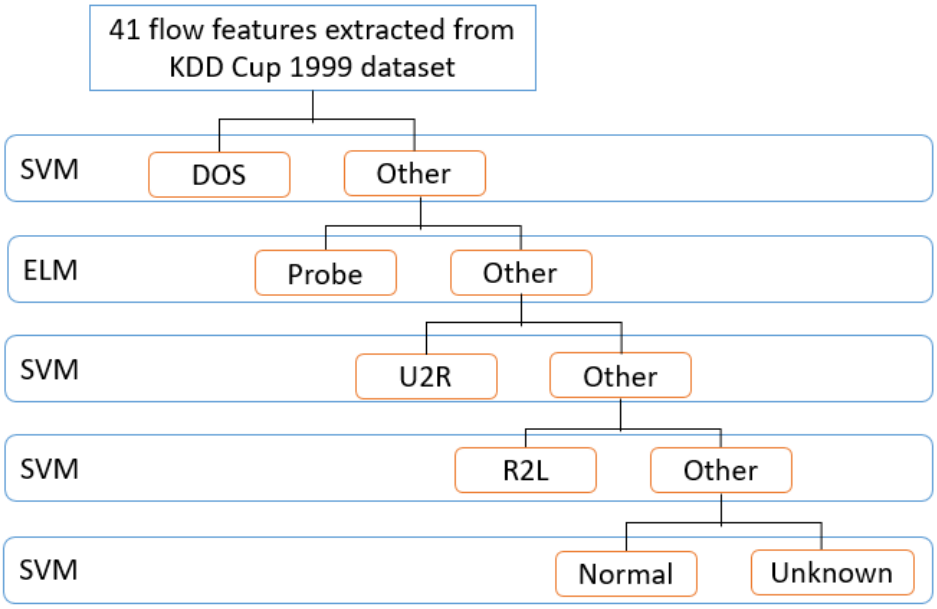

Figure 4. Multi-level hybrid IDS via class imbalance with deep learning.

Rani [24] developed an IDS using DNN with the classifier-level class imbalance approach. The network data are first preprocessed using data conversion, before being normalized using the min–max method. The normalized information is then sent into a neural network, which modifies the cross-entropy function to address the problem of class imbalance. It is accomplished by weighing the classes while the classifier is being trained. The system achieved 85.56% accuracy on the NSL-KDD dataset. Imrana et al. [25] developed an IDS using a bidirectional Long-Short-Term-Memory (BiDLSTM). The efficiency of the BiDLSTM technique was validated on the NSL-KDD dataset and outperformed the standard LSTM. It achieved 91.93% accuracy, while standard LSTM achieved 87.26%.

Chen et al. [26] developed a network anomaly detection model based on clustering, followed by a density peaks clustering technique, DPC-GS-MND, based on grid screening and mutual neighborhood degree. They achieved 96.83% accuracy on the KDDCUP99 dataset. Ramadan et al. [27] developed an IDS for Flying Ad Hoc Networks based on Recurrent Neural Networks (RNN), and achieved 91% on the KDDCUP99 dataset. Table 1 summarizes the related work, including the utilized technique, dataset, and the achieved accuracy.

Table 1. Related Work Summary.

Study	Technique	Dataset	Accuracy (%)
Tang et al. [15]	Simple DNN	NSL-KDD	75.75
Rawat et al. [16]	DNN	NSL-KDD	75.9
Wang et al. [17]	Semi-Supervised Approach	NSL-KDD	77.26
Dey et al. [18]	Random Forest	NSL-KDD	81.95
Latah and Toker [19]	Decision Tree	NSL-KDD	88.74
Tang et al. [20]	GRU-RNN	NSL-KDD	89
Latah and Toker [21]	KNN, ELM, and HELM	NSL-KDD	84.29
Gao et al. [6]	A-PCA-I-ELM	NSL-KDD	81.22
Zheng et al. [22]	ILECA	NSL-KDD	92.35
Al-Yaseen et al. [23]	SVM and ELM	KDDCUP99	95.75
Rani [24]	Classifier-level DNN	NSL-KDD	85.56
Imrana et al. [25]	LSTM, BiDLSTM	KDDCUP99	87.26, 91.36
Chen et al. [26]	DPC-GS-MND	KDDCUP99	96.83

Ramadan et al. [27]	LSTM-RNN	KDDCUP99	91
Chung and Wahid[28]	Simplified Swarm Optimization	KDDCUP99	93.3
Ambusaidi et al. [29]	Least Squares SVM	KDDCUP99	92.8
Khalvati et al. [30]	SVM	KDDCUP99	94.8
Mohammadi et al. [31]	FGLCC, FGLCC-CFA	KDDCUP99	92.59, 95.05
Alazzam et al. [32]	Sigmoid _PIO, Cosine_PIO	KDDCUP99	94.7, 96

3. Smart City

Many security and privacy concerns have emerged due to the rising use of intelligent apps. The smart city network includes a wide range of devices due to new technology, and a single hacked item might render the whole collection susceptible; exploiting such flaws enables hackers to start a series of cyber-attacks. For a city to adopt innovations into smart city cyberinfrastructure and to enhance the living circumstances of its residents, smart city security is critical. Availability, confidentiality, integrity, access control, privacy, and non-repudiation are some of the primary security and privacy criteria. The discovered data of a smart city's physical areas contains precise facts regarding the people who dwell in such surroundings [33].

Sustainability means meeting present needs without risking future generations' abilities to meet their own. The sustainable development goals (SDG) include cities where everyone has access to smart services, electricity, housing, transportation, and other amenities. The relations between SDGs and smart cities can be listed as follows:

- Ensure everyone has access to affordable, safe, and appropriate housing and services, and improve slums.
- Ensure everyone has access to sustainable, accessible, cheap, and safe transportation systems, focusing on road safety and vulnerable populations such as the elderly, persons with disabilities, children, and women.
- All nations should be able to design and manage human settlements in a participatory, integrated, and sustainable manner.
- Enhance global efforts to conserve and to protect the world's heritage.
- Assist LDCs in creating sustainable and resilient structures with local resources, including financial and technical assistance.

Figure 5 shows that every aspect of a smart city can be utilized to achieve SDGs as follows:

- Smart Mobility includes integration with ICT, providing clean and non-motorized options.
- Smart People: This includes a society that encourages creativity, inclusion, and smart education.
- Smart Living: This is achieved by providing a healthy, safe, and happy life.
- Smart Economy: This is achieved by encouraging innovation, entrepreneurship, productivity, and interconnection.
- Smart Government: The government should have a transparent policy, open data, and provide e-governance.
- Smart Environment: This includes environments that have green buildings, energy, and planning.

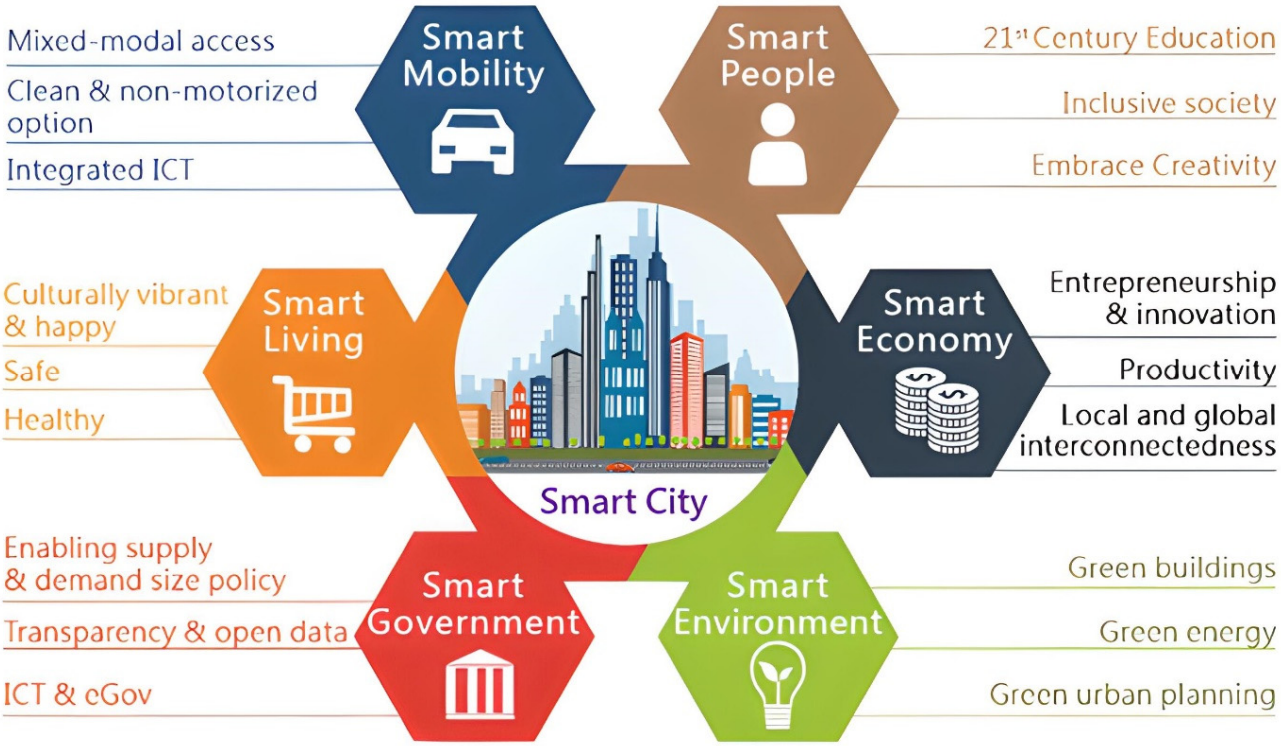

Figure 5. SDG and Smart City.

The introduction of sustainable technological developments to the network of smart cities is expected to enhance people's quality of life by improving the functioning of urban systems and promoting sustainable development. The advent and deployment of this technology in various intelligent systems has made security and privacy problems a key challenge that needs robust solutions. Due to the expanding use of AI, AI systems are critical in multiple smart applications, including autonomous home appliances, pacemakers, and system control. For instance, providers and device manufacturers may use data mining technology to collect and analyze sensitive information and personal data, and achieve various essential service-related goals. Additionally, attackers who comprehend AI are themselves more intelligent. Certain cybercriminals can examine how ML defenses are developed, and are trained to deploy specific approaches to degrade algorithm reliability. However, significant security hazards are associated with the increased usage of AI technology. Since new technologies are integrated into smart city networks, edge-based structures face new security concerns, as edge-dispersed operating settings are more susceptible to assaults than centralized clouds [34]. As a result, a system that can detect and prevent attacks on smart city edge devices is needed.

Smart city systems, comprising sensor devices that are scattered throughout dangerous areas, provide a plethora of security risks, and need security risk management and mitigation. It is critical to develop techniques for mitigating such hazards to enable the deployment of smart city apps, and to boost users' desire to utilize them. As a result of the diversity of sensor networks and gadgets in smart cities, developing a risk management plan is a significant challenge.

The term "sustainable smart city network" refers to a network that allows smart cities to connect; this notion was a relatively recent trend in the mid-2010s [35]. Smart cities, urbanization, sustainability, and ICTs are examples of the same idea. According to Mohanty et al. [36], technical criteria for a sustainable smart city necessitates the integration of numerous qualities. The bulk of ideas for smart city building were four primary characteristics: sustainability, smartness, quality of life, and urbanization. The following are some of the advantages of incorporating smart city traits:

- Sustainability: The ability to help a city reach ecological balance while maintaining and operating the city.
- Smartness: Aspirations to improve the city's citizens' environmental, economic, and social situations.
- Quality of life: Nowadays, we can assert that an urban citizen's financial and emotional well-being reflects an increase in their quality of life, with citizens catalyzing urban growth; these solutions seek to enhance educational opportunities, housing quality, health conditions, and social cohesion.
- Urbanization: Distinctive urbanism is centered on economic, technical, infrastructural, and governance elements of the transition from a rural to an urban environment.

The ML-based IDS for a smart city has three main components. Initially, the endpoint layer is the layer of smart devices that operate in the Internet environment, such as smart water, energy, and traffic, etc. Next, the intermediate layer is the controller, such as the router that connects with the smart devices closest to the endpoint. The middle layer uses a deep-run intrusion detection algorithm to evaluate traffic from smart devices, and classify it as regular or abnormal. If the traffic is normal, it is forwarded to the next component. The highest layer connects cloud nodes by a specified scale of fog nodes [37]. This paper aims to build the IDS to run in the intermediate node to identify abnormal traffic and to prevent it from accessing smart city edge devices.

4. Classification

Classification is accomplished via the use of a classifier algorithm. The critical element is categorization action that is associated with learning, and which is often referred to as classification techniques. ML is a method of representational learning that focuses on the evolution of systems that perform AI tasks such as prediction, diagnosis, and recognition, etc. The three main forms of learning are supervised, unsupervised, and semi-supervised [38].

ML algorithms have three main types: supervised, unsupervised, and semi-supervised learning. Deep learning is a promising learning scheme in ML [39]. Guided categorization, often known as supervised learning, is a kind of learning that enables the recognition of a set of likely classes in advance. Correctly categorized data examples are used as a training set in this procedure. In the following phase, supervised learning algorithms anticipate the outcome.

Consequently, this is an excellent solution for when a certain target value is needed. This kind of learning also requires a set of training datasets that comprise both the input and the anticipated outcomes. The training set is then used to classify new data. SVMs, ANNs, logistic regression, NB, KNN, RF, DT, and other supervised learning methods are considered to be classification algorithms [40]. Descriptive or undirected categorization refers to unsupervised learning. The term "unsupervised" refers to data that does not have an expected outcome. Clustering, self-organizing map (SOM), and deep learning, are well-known unsupervised learning methods. Semi-supervised learning uses labeled and unlabeled data to combine supervised and unsupervised learning [41]. The NB, SVMs, ANNs, DT, ensemble learning, and ELM are discussed. A comparison of these algorithms is in Table 2.

- **Naïve Bayesian**

The NB classifier is a primary probabilistic classifier that is often used to identify network intrusions. It combines previous knowledge with sample data and utilizes statistical deductions, which use probability to show various sorts of uncertainties. The concept that all input attributes are uncorrelated to one another forms the basis of its principles. As a result, it shows how likely it is that an object is from a specific class. From Tao et al. [42], in order to use the time-NetFlow link, the NB method was paired with a time slicing

function, as network traffic fluctuates at various periods and some traffic does not appear at all.

- **Support Vector Machines (SVMs)**

A SVM is used to look for patterns. It uses use supervised learning ideas such as kernels (e.g., radial basis function (RBF)), sparseness, the lack of local minima, and capacity checking accomplished by acting on the boundary (the hyperplane solution's distance from its nearest point). In the learning domain, classifiers with a high degree of generalization correctly predict the class of new input [43]. From Kabir et al. [44], the least squares support vector machine has been proposed as a basis for an IDS, a basic SVM classifier (LS-SVM) variation. Compared to a standard SVM, this modification is more prone to be influenced by outliers and noise in the training set. There are two steps to their decision-making process. The first step is responsible for shrinking the dataset to a manageable size using an optimum allocation approach that selects samples based on data variability. The LS-SVM is then fed these representative samples in the following step. This technique was designed to function with incremental and static data, which yielded good performance.

- **Artificial Neural Networks (ANN)**

ANNs are mathematical representations of the structure of the brain of intelligent creatures that learn via experience. In reaction to inputs and responses from the environment in which they function, they are capable of self-adaptation, self-organization, and learning. While NNs are inspired by biological design, they are mainly used as classifiers in anomaly detection. The most often utilized ANN techniques are Back Propagation (BP) and Multi-layer Perceptron (MLP) [45]. Brown et al. [46], depending on the protocols used at the application layer, such as FTP, SMTP, and HTTP attributes, proposed a two-class classifier based on an evolutionary general regression NN (E-GRNN) as an IDS. The authors employed evolutionary computing to discover the best configuration for the general regression neural network, by modifying parameters and distinguishing characteristics (feature mask). By removing unneeded information, this technique decreases computing complexity while also improving classification accuracy. Only labeled data, which are difficult to obtain, owing to the need for expert knowledge, may be used to train a classifier with the assistance of supervised learning models. Unsupervised approaches, on the other hand, only analyze the unlabeled data, which is widely available in real-world circumstances.

- **Ensemble Approach**

When opposed to employing individual classifiers, an ensemble technique combines the outputs of many classifiers into a unified one, resulting in improved performance [47]. Bukhtoyarov and Zhuckov [48] have proposed a unique tree-level strategy for merging individual classifier choices to develop an ensemble distributed classifier for networks. This approach is based on using Genetic Programming (GP)-based Ensembling (GPEN). GPEN develops a program that shows how to use GP operators to combine component network forecasts to obtain an ensemble forecast.

- **Decision Tree (DT)**

Each node in the DT may compare and contrast different actions based on their costs, benefits, and probability. It is a diagram representing the probable consequences of a set of connected decisions [49]. Typically, a DT starts with a single node and then branches out to include more options. Each of these occurrences results in the addition of nodes, which results in the acquisition of instances. As a consequence, it takes on the shape of a tree.

Table 2. Comparison of classification algorithms [50].

Model	Advantages	Disadvantage
Ensemble approach	•Enhances precision and stability. •Reduced variance, contributing to the avoidance of overfitting problems.	•Complexity of computation. •Difficult to interpret if the model is large. •Requires fine-tuning of various parameters.
NB Classifier	• It performs well with textual data. • It is simple to build. • It is quick when compared to other methods.	• A fundamental assumption regarding the data distribution's shape. • Due to a lack of data, a frequentist must estimate a probability value for all potential values in the feature space.
SVM	• SVMs may be used to describe decision boundaries that are not linear. • When linear separation is required, it performs comparably to logistic regression. • SVM is resistant to overfitting concerns.	• A vast number of dimensions contribute to the results' lack of transparency. • Selecting an effective kernel function is difficult (prone to overfitting/training difficulties). • Memory complexity.
DT	• Decision trees are a rapid approach for both learning and prediction. • They are well-suited for handling qualitative (categorical) data. • They work best with decision boundaries parallel to the feature axis.	• Problems with diagonal decision boundaries. • Easy to overfit. • Extremely sensitive to tiny data perturbations. • Out-of-sample prediction issues.
ANN	• Recognize complicated connections between dependent and independent variables with ease. • Capable of dealing with noisy data.	• Local minima. • Overfitting. • The processing of an ANN network is difficult to understand and takes a long time.
ELM	• Fewer optimization restrictions. • Increased efficiency. • Simple implementation.	Poorly conditioned hidden layer output matrices lead to low robustness.

5. Research Methodology

5.1. The Proposed System

As shown in Figure 6, our proposed system is composed of three components. First is the traffic coming from smart city terminal nodes. The second is the anomaly classification system that takes traffic as the input from the first component. Using a proposed IDS approach, it evaluates traffic from smart devices and classifies it as being regular or anomalous. This multi-classification system can detect if the packet is normal or an attack, and the type of attack. The third is the cloud layer that takes the output from the second component. It communicates with cloud nodes via connecting to a set of fog nodes. If the IDS detects normal traffic, the traffic is allowed to communicate with the cloud system, otherwise, the traffic is prevented.

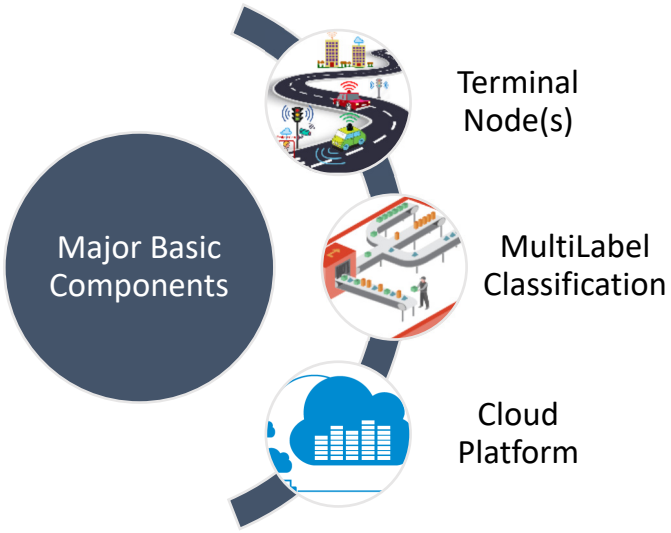

Figure 6. The proposed system components.

SALMA is a five-tier classification model based on ELM algorithms presented to categorize anomalies. As shown in Figure 7, R2L and DOS assaults are identified first, since their detection accuracy is the greatest. Then, both Probe and U2R attacks are in the next levels, as their models have lower accuracies. Each layer in our model has a single classifier. The training dataset is separated into two categories at each layer: one for each form of attack, and another for "Other" traffic, or an assault that the next layer must identify. Consequently, the model is trained on five datasets and it adopts a one-versus-all technique. We utilize the ELM technique in all layers, because of its resilience against noisy training data.

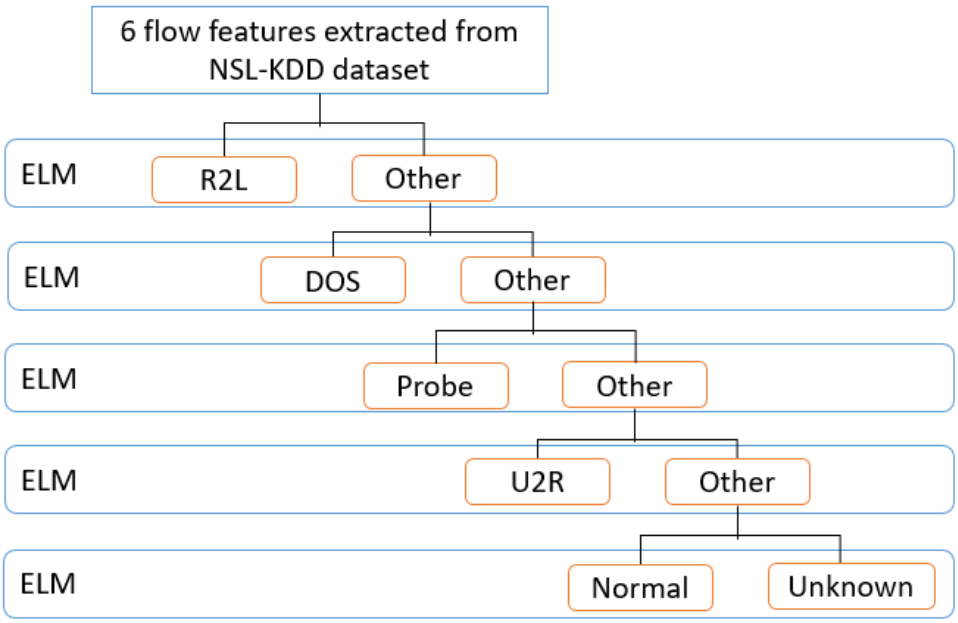

Figure 7. The proposed system architecture.

5.2. ELM

Professor Guang-bin Huang devised the ELM in 2004, A single-hidden-layer NN that is very generalizable, needs little human interaction, and that assures the assumption of a specific learning accuracy; the algorithm's speed is greatly boosted [51]. It saves substantial time and money when compared to traditional NNs. The approach does not need frequent adjustments to the weights and hidden units throughout the training stage. While each neuron has an offset, random initialization determines the hidden layer and input weight offsets. To finish the training, we only need to know how many hidden layer neurons were used throughout the training; we can then extract the model's output weight.

Assume that you have N training samples given for a single hidden layer feed-forward NN (x_i, t_i), where $x_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in R^n$ is the sample input vector and $t_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \in R^m$ is the intended output vector, where n denotes the number of features contained in the input sample, and m is the training sample's total number of classes. Additionally, since this single hidden layer neural network includes L hidden nodes, the output of the network may be expressed as follows [52]:

$$\sum_{i=1}^{L} \beta_i g(\omega_i . x_j + b_i) = o_j, j = 1,2, \ldots, N \quad (1)$$

$g(\bullet)$ denotes the activation function, $\omega_i = [\omega_{i1}, \omega_{i2}, \ldots, \omega_{in}]^T$ is the input weight vector between the input layer and the hidden layer, and $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{in}]^T$ is the output weight vector between the hidden layer and the output layer; b_i is the ith offset value of the hidden layer's first node, where $\omega_i . x_j$ indicates the weight assigned to the inner product of the value and the value of the training sample, and o_j signifies the network model's actual output. Figure 8 shows the single hidden layer feed-forward network ELM model.

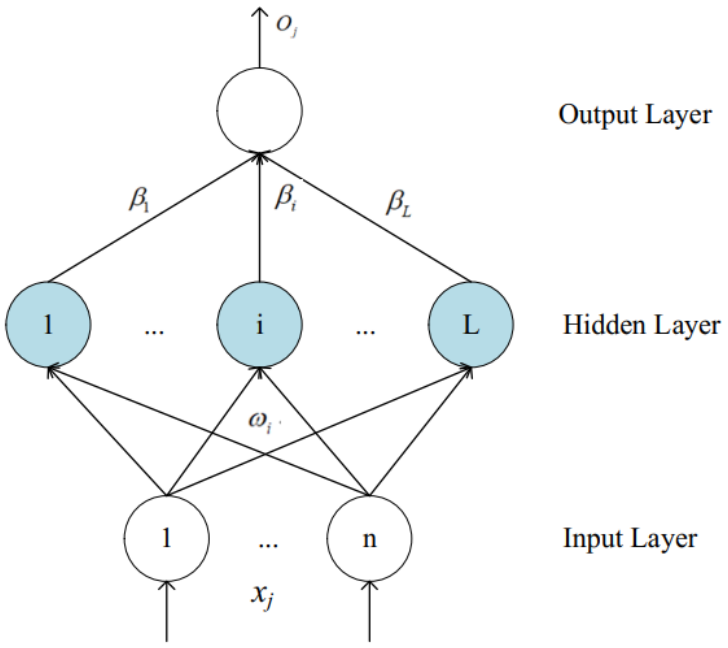

Figure 8. Feed-forward network with a single hidden layer [52].

The single hidden layer feed-forward NN's objective is to minimize the output result's error value, which is as follows:

$$\sum_{j=1}^{N} \left\| o_j - t_j \right\| = 0 \tag{2}$$

where t_j denotes the expected result. β_i, ω_i, and b_i exist according to Formulae (1) and (2) to make the following formula accurate [52]:

$$\sum_{i=1}^{L} \beta_i g\left(\omega_i . x_j + b_i\right) = t_j, j = 1,2, \ldots, N \tag{3}$$

Equation (3) may be simplified as follows, based on the matrix:

$$H\beta = T \tag{4}$$

H is the hidden layer output value matrix, β, is the hidden layer to the output layer weight matrix, and T denotes the projected output matrix in the formula. H, T, and β are further stated as *follows* [52]:

$$H = \begin{bmatrix} g(\omega_1 . \mathrm{x}_1 + \mathrm{b}_1) & \cdots & g(\omega_L . \mathrm{x}_1 + \mathrm{b}_\mathrm{L}) \\ \vdots & \ddots & \vdots \\ g(\omega_1 . \mathrm{x}_\mathrm{N} + \mathrm{b}_1) & \cdots & g(\omega_L . \mathrm{x}_\mathrm{N} + \mathrm{b}_\mathrm{L}) \end{bmatrix}_{N \times L} \tag{5}$$

$$\beta = \begin{bmatrix} B_1^T \\ \vdots \\ B_L^T \end{bmatrix}_{L \times M} \tag{6}$$

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_L^T \end{bmatrix}_{N \times M} \tag{7}$$

In the vast majority of circumstances, $H\beta = T$ cannot be demonstrated. For training a model, several criteria are discovered: β_i, ω_i, and b_i, where $i \in [1 \ldots L]$, and where L is the number of layers. These criteria need to be changed so that the error value is as minimal as possible, as shown in the following equation:

$$\left\| H(\omega_i, b_i)\beta_i - T \right\| = \min_{\omega_i, \beta_i, b_i} \left\| H(\omega_i, b_i)\beta_i - T \right\| \tag{8}$$

When faced with such challenges, a typical NN approach would progressively increase the parameters throughout the iteration phase, resulting in model's training period being extended. The hidden layer's input weights and bias are randomly initialized throughout the Extreme Learning Machine's model training. A weight matrix for the output is also provided. Because H is known, the model becomes a linear system, $H\beta = T$, which can be solved using least-squares β. The following is the formula:

$$\beta = H^{+}T \tag{9}$$

The symbol H^{+} denotes the Moore–Penrose generalized inverse matrix of H's output weight matrix in Equation (9). To summarize, the ELM has the following learning process:

Input: N training samples (x_i, t_i), $x_i \in R^n$, $t \in R^m$, $i = 1,2, \ldots, N$.

Output: The output weight β from the hidden layer to the output layer.

I. Initialize the weights of the inputs I and the offset of the hidden layer b_i at random;
II. Determine the hidden layer H's output weight; and
III. Calculate the output weight from the hidden layer to the output layer β.

5.3. ELM Hyperparameters

The following are the hyperparameters of ELM (the best hyperparameters are presented in Table 3):

- The number of nodes for our hidden layer, i.e., hiddenSize (L);

- Input weight W; and
- bias 'b';
- Activation functions.

Based on a weighted sum of inputs, activation functions are employed to calculate the output response of neurons. Throughout the network, each layer uses a single activation function. It is available in two variants: linear and nonlinear activation functions. Nonlinear activation functions are often employed to solve classification problems. The following are some of the nonlinear activation functions [53]:

- **Sine Function**: It accepts a real number and returns another real value that ranges between 1 and −1.
- **Hard Limit Function**: It is a value-assigning limiting function with a threshold. A value of 0 or 1 is assigned to each neuron location. When it reaches the threshold, it returns 1; otherwise, it returns 0.
- **Triangular Bias Function**: The limit of triangular inclination may function as a neuronal exchange. This limit defines the yield of a layer based on its known data.
- **Radial Bias Function**: It is a function proportional to the distance to the origin.
- **Sigmoid Function**: It's a 'S'-shaped activation function with the formula F(x) = 1/1 + exp(−x), with values ranging from 0 to 1.

Table 3. ELM Hyperparameters [54].

Variable Name	Available Values	Best-Value
hiddenSize.	1, …, inf	2000
Activation Function	Sine, Sigmoid, RBF, Triangular Bias, Hard Limit	Sigmoid
Cost parameter	1, …, inf	2^6

6. Experimental Results and Discussion

6.1. PC Properties

The categorization experiments were conducted using a Python-based software package on an Intel i7 computer with 8 GB of RAM.

6.2. Dataset Characteristics

The NSL-KDD dataset [14], a scaled-down version of the original KDDCUP99 dataset [55], has become a gold standard for testing IDSs. In addition to a class attribute, there are 41 attributes in total. The training dataset has 21 different assaults, whereas the test dataset contains 37 different types of attacks. Table 4 describes four types of attacks: probe, R2L, U2R, and DoS.

- Probe—Before initiating an attack, the attacker acquires knowledge regarding the various faults in the target system.
- Denial of Service (DoS) attack—Once an attacker tries to use computational resources to increase bandwidth or to overwhelm a device service, legitimate users are barred from accessing it.
- User to root attacks (U2R)—After gaining access to a local target host, an attacker attempts to get super or root permissions on the device.
- Remote to user attacks (R2L)—An attacker tries to get into a victim's system or network without a legitimate account.

Table 4. KDDCUP99 Attack Types [55].

Attack Category	Attacks (37)
DoS	Back, Edstrom, Smurf, Worm, Mailbomb, Apache2, Land, Pod, Process table, Mailbomb Neptune Teardrop
Probe	IPsweep, Mscan, vPortsweep, SaintI, Satan, Nmap
R2L	Ftp_write, Sendmail, Snmpgetattack, Xsnoop, Waremaster, Snmpguess, imap, Httptunnel, Named, Phf, Xlock, Multihop Guess_password
U2R	Xterm, Sqiattack, Buffer_overflow, Loadmodule, Ps, Perl, Rootkit

The NSL-KDD dataset contains the attributes shown in Table 5. A1, A2, A5, A6, A23, and A24 are characteristics that may be readily accessed from the SDN controller [15].

Table 5. NSL-KDD Attributes [21].

A. #	Attribute Name	A. #	Attribute Name	A. #	Attribute Name
A1	Duration	A15	Su attempted	A29	Same srv rate
A2	Protocol type	A16	Num root	A30	Diff srv rate
A3	Service	A17	Num file creations	A31	Srv diff host rate
A4	Flag	A18	Num shells	A32	Dst host count
A5	Source bytes	A19	Num access files	A33	Dst host srv count
A6	Destination bytes	A20	Num outbound cmds	A34	Dst host same srv rate
A7	Land	A21	Is host login	A35	Dst host diff srv rate
A8	Wrong fragment	A22	Is guest login	A36	Dst host same src port rate
A9	Urgent	A23	Count	A37	Dst host srv diff host rate
A10	Hot	A24	Srv count	A38	Dst host serror rate
A11	Number failed logins	A25	Serror rate	A39	Dst host srv serror rate
A12	Logged in	A26	Srv serror rate	A40	Dst host rerror rate
A13	Num compromised	A27	Rerror rate	A41	Dst host srv rerror rate
A14	Root shell	A28	Srv rerror rate	A42	Class label

6.3. Performance Measures

The selection of a Network Anomaly Detection System (NADS) for a particular environment is a wide topic that may be summarized as an IDS evaluation [56]. The performance metrics to consider while deploying/choosing an IDS/anomaly detection system are discussed in the following, presenting how these metrics are used to evaluate the approach presented. True Positives (TP) are accurately identified as attacks, whereas True Negatives (TN) are normal connections that are correctly identified as normal connections. Table 6 provides information on the performance metrics used to evaluate IDSs [57].

Table 6. IDS Performance Measures [58].

Performance Metric	Description	Formula
Detection Rate/Precision	The proportion of true positives among projected positives (or) proportion of test data flagged as an attack that is an attack.	TP/(TP + FP)
Accuracy	To establish the total accuracy, take a measurement. It is the percentage of accurately predicted values over the whole dataset.	TP + TN/(TP + FP+ FN +TN)
False Alarm Rate	The false-positive rate (FPR), also known as the false alarm rate (FAR), is the percentage	FP/(FP + TN)

	of legitimate packets mistakenly identified as malicious.	
True Positive Rate Sensitivity/Recall	The proportion of attack classes successfully detected (or) the percentage of true positives projected as positives.	TP/(TP + FN)

The F-score/F-measure is a mathematical expression representing the harmonic mean of accuracy and recall, as shown in Equation (14) [59]. The F_score is a metric for determining a test's accuracy [60]. Improved accuracy and recall are required for good IDS performance [61]. When calculating the F_score, both accuracy and recall are taken into account [58]:

$$F\_Score = \frac{2 * Precison * Recall}{Precision + Recall} \quad (10)$$

6.4. Visualization

In this section, we are making some visualization of our proposed system. Figure 9 shows the distribution of the NSL-KDD dataset [14] across the different labels, where the x-axis is the label category, which is either the normal or attack category (DOS, R2L, U2R, or Probe), and the y-axis is the count of samples in each category in the dataset. It is observed that the dataset is somewhat balanced between normal and attack, while the attack categories are not balanced, where DOS presents the most commonly observed category in the dataset, and U2R is the least observed one.

In Figure 10, for binary classification between normal and attacks, which are referred to as attack_flag in the figure, all the features have a positive correlation except for protocol_type. Count, duration, src_bytes, and dst_bytes positively correlate with attack_flag in descending order. The binary classification between DOS and the others is called dos_flag in the figure. Count and srv_count have a positive correlation with dos_flag in descending order. Duration, src_bytes, dst_bytes, and protocol_type negatively correlate with dos_flag. The binary classification between Probe and others is referred to as probe_flag in the figure. protocol_type, count, and srv_count have a negative correlation with probe_flag. Duration, src_bytes, and dst_bytes have a positive correlation with probe_flag. For binary classification between U2R and others, this is referred to as u2r_flag in the figure. All of the features except for protocol_type have a negative correlation with u2r_flag. For binary classification between R2L and others, this is referred to as r2l_flag in the figure. protocol_type, count, and srv_count have a negative correlation with r2l_flag. Duration, src_bytes, and dst_bytes has a positive correlation with r2l_flag. For multi-classification between normal, Probe, DOS, U2R, and R2L are called attack_map. The protocol_type and srv_count have a negative correlation with attack_map. Count, duration, src_bytes, and dst_bytes have a positive correlation with attack_map.

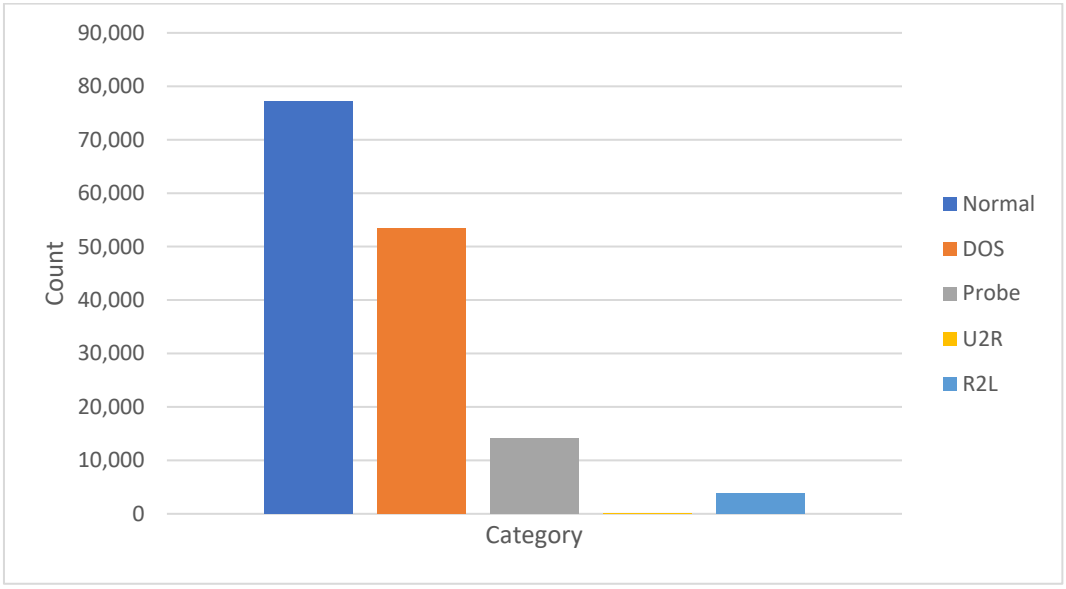

Figure 9. Dataset Label Distribution.

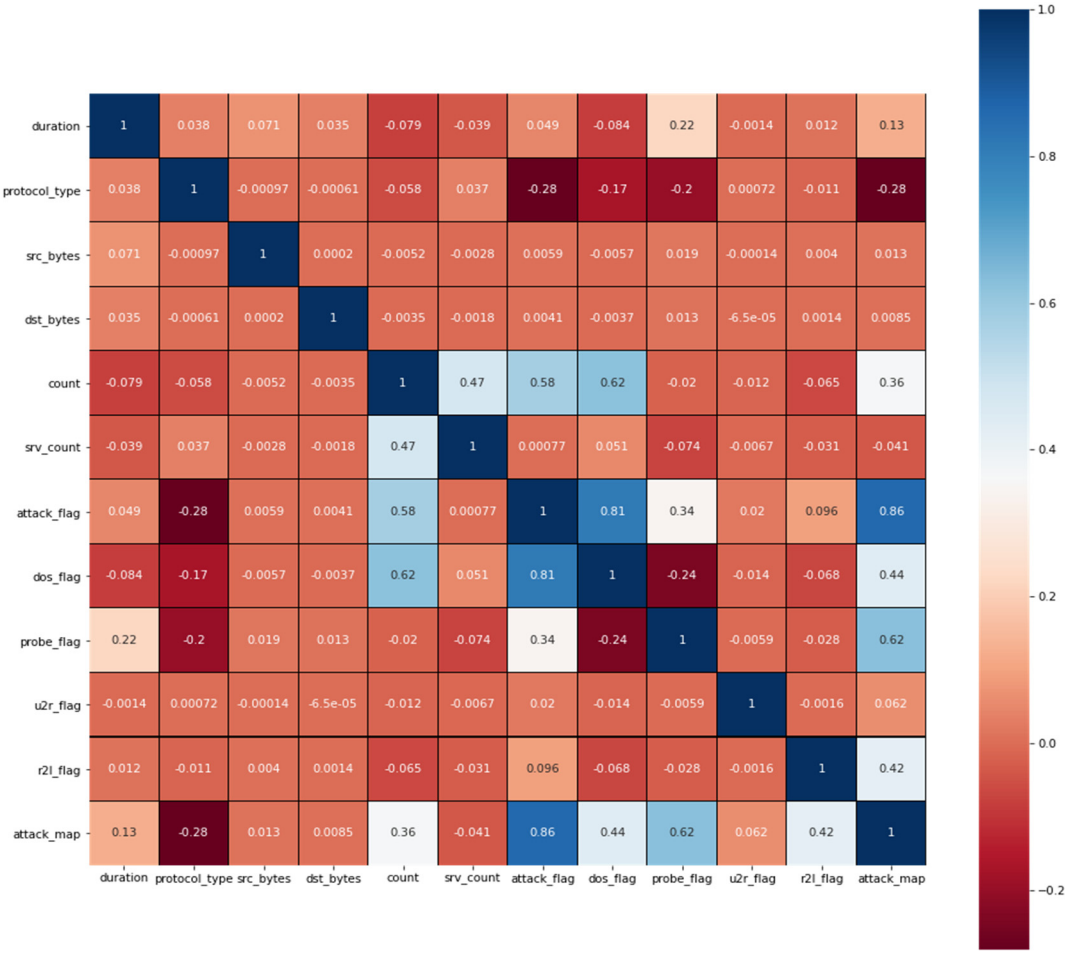

Figure 10. Heatmap illustrating feature importance and correlation.

7. Comparative Analysis and Discussion

The NSL-KDD dataset is used in Table 7 to compare the proposed system to existing state-of-the-art approaches. We compared our proposed system to other previously mentioned ML techniques [5,13–20,22,23]. Related to Table 7, our system is highly accurate

and has a low percentage of false alarms. The model in [15] is a straightforward DNN with just three hidden layers. The parameters of this system need to be tuned in order to achieve better results. The model in [16] was also a DNN, and it achieved almost the same results as the model in [15]. The authors [17] used a semi-supervised approach when the model was trained exclusively on a tiny training set by self-evolution. It was trained on fewer data, which led to other approaches performing better than it. In [18], the model used multiple feature selectors to achieve the best result, using a gain ratio that selected nine features. Training the model with more features would lead to better results. The model in [19] earned the highest accuracy and F1-score; nevertheless, they used PCA as a feature selection approach to optimize the system's overall performance.

Additionally, the study in [19] demonstrates that PCA may outperform techniques such as information gain, which is widely utilized for feature selection. The model in [20] is GRU-RNN, with just three hidden layers. The model parameters in [20] need to be tuned in order to achieve better results. In [21], the authors used the multi-level classification approach used by the proposed approach, and achieved a high level of precision. They applied KNN, ELM, and HELM in their models. The model in [6] used A-PCA-I-ELM, but it used all 41 features and thus achieved good results. The model in [22] used ILECA, where the dimensionality was reduced with ELM. Using more features may lead to better results. In [25], the authors used conventional LSTM and BiDLSTM. BiDLSTM achieved better results. The model achieved the best f-score and false alarm rate, but it used all of the features. In [24], the best detection rate was achieved by solving the class imbalance problem using oversampling techniques, and then DNN was used for classification. Figure 11 shows a comparison in terms of accuracy between SALMA and previous state-of-the-art techniques.

Table 7. Comparison between the proposed system and previous approaches to the NSL-KDD dataset.

Method	Accuracy (%)	False Alarm Rate (%)	Precision (%)	Recall (%)	F1-Score (%)
Simple DNN [15]	75.75	3.21	92.50	59.95	74.13
Semi-Supervised Approach [17]	77.26	N.A	N.A	N.A	N.A
Random Forest [18]	81.95	N.A	N.A	N.A	N.A
Decision Tree [19]	88.74	3.99	83.24	**96.5**	**89.38**
GRU-RNN [20]	89	N.A	89	89.5	89.2
KNN, ELM, and HELM [21]	84.29	6.3	**94.18**	77.18	84.83
DNN [16]	75.9	N.A	N.A	N.A	N.A
Classifier-level DNN [24]	85.56	N.A	**97.09**	76.94	85.85
LSTM [25]	87.26	4.03	90.34	87.26	88.03
BiDLSTM [25]	91.36	**0.88**	92.81	91.36	**91.67**
ILECA [22]	92.35	N.A	N.A	N.A	N.A
A-PCA-I-ELM [6]	81.22	N.A	**96.1**	N.A	N.A
Proposed System (SALMA)	**95.04**	**2.48**	92.05	79.59	85.37

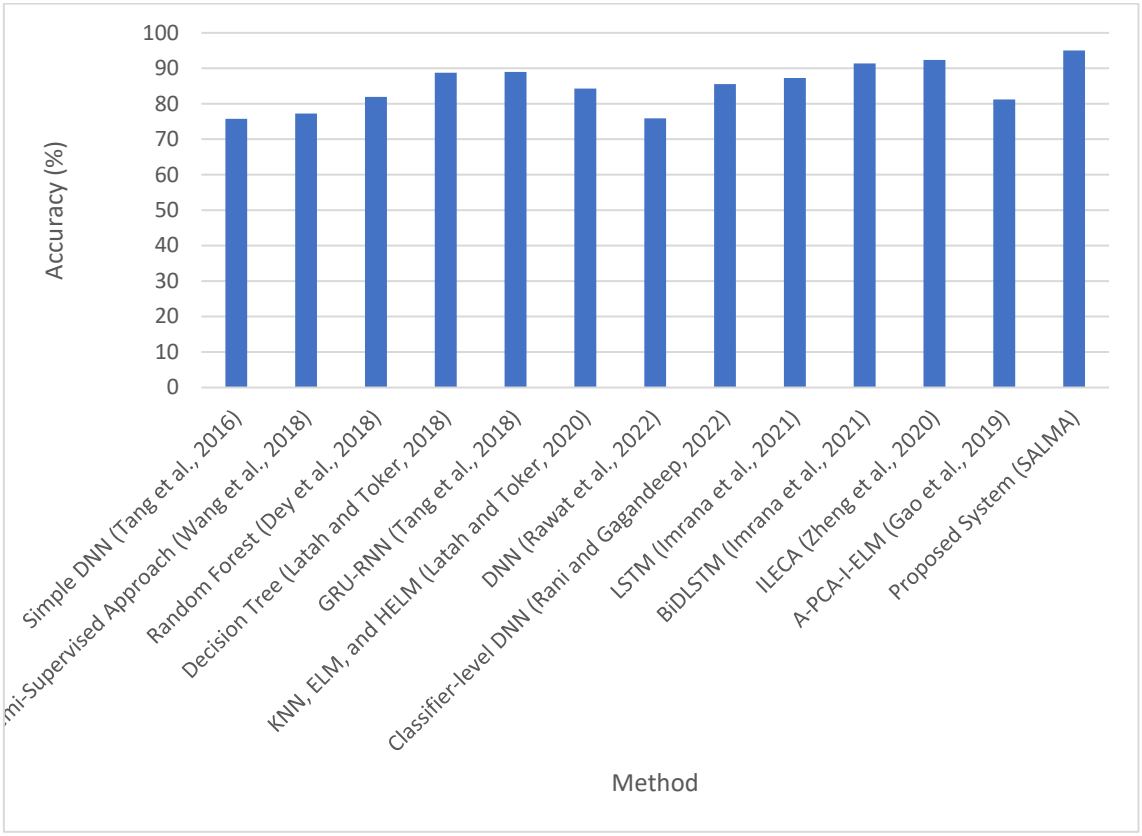

Figure 11. Accuracy comparison between the proposed system and previous approaches on the NSL-KDD dataset [5,13–20,22,23].

Table 8 compares the **SALMA** and other state-of-the-art methods, using the KDDCUP99 dataset [55]. Our proposed system was compared to different Machine Learning approaches [21,24–30]. Our approach achieved the best accuracy. In [28], the authors used Simplified Swarm Optimization (SSO) guided via Weighted Local Search (WLS). Other optimization functions may achieve better results. In [29], mutual information is used as a feature selector fed to the ML model. Other feature selectors may achieve better results. In [30], a genetic algorithm selected 10 features before applying the SVM model that achieved relevant results. In [31], Filter-based grouping linear correlation coefficient (FGLCC) and a combinatory approach of FGLCC with cuttlefish (FGLCC-CFA) were used and achieved good results. In [32], the "Pigeon Inspired Optimizer (PIO)" was proposed as a new method, where the "cosine" and "sigmoid" transfer functions were used to find the fitness function of the optimization process, and achieved good results. In [23], the authors used a multi-level classification approach where SVM and ELM were used according to their approach, and achieved high accuracy. In [26], DPC-GS-MND achieved the best results after SALMA, but it used all the features. In [27], RNN was used, but the techniques did not achieve high results as RNN is best suited for sequence models. Figure 12 compares the accuracy between the proposed system and other state-of-the-art methods.

Table 8. Comparison between the proposed system and previous approaches on the KDDCUP99 dataset.

Method	Accuracy (%)	No. of Features
Simplified Swarm Optimization [28]	93.3	6
Least squares SVM [29]	92.8	6
SVM [30]	94.8	10
FGLCC-CFA [31]	95.05	10
FGLCC [31]	92.59	16

Sigmoid _PIO [32]	94.7	10
Cosine_PIO [32]	96	7
SVM and ELM [23]	95.75	6
DPC-GS-MND [26]	96.83	41
LSTM-RNN [27]	91	41
Proposed Approach (**SALMA**)	**99.24**	6

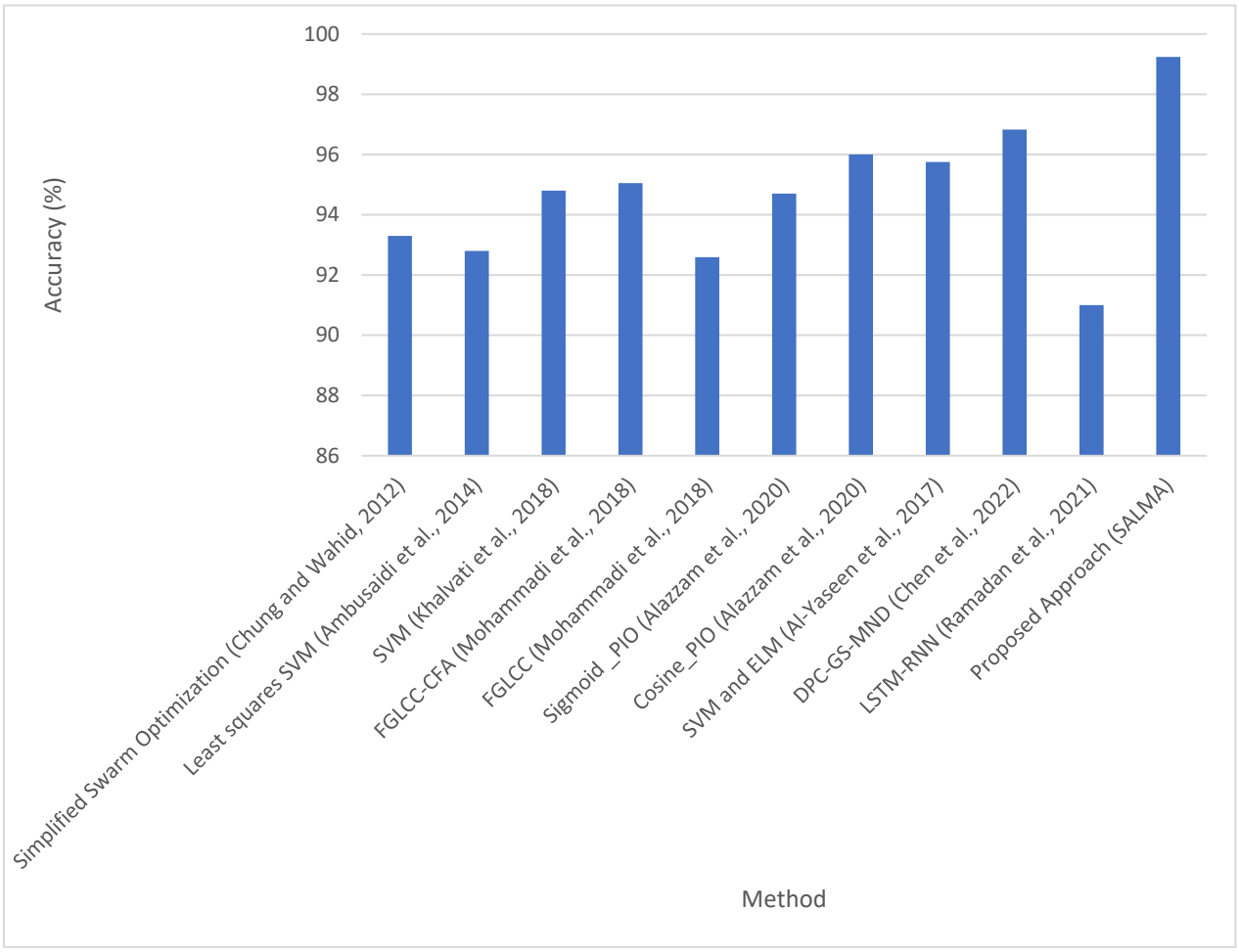

Figure 12. Accuracy comparison between the SALMA and previous approaches on the KDDCUP99 dataset [21,24–30].

8. Conclusions and Future Work

This study examines the notion of a smart-cities security IDS for effectively upgrading a typical IDS for smart city IoT applications. In this work, we proposed **SALMA** as a multi-level hybrid IDS for SDNs based on ELM that is used to develop a smart attack learning machine advisor system for protecting smart cities from smart threats. The system is based only on six flow features used in SDNs. The experimental research using the NSL-KDD dataset revealed that our methods considerably increased the overall accuracy compared to typical supervised learning algorithms (ELM). Furthermore, with a 95% accuracy rate, the approach recognized the new attacks included in the testing set. Additionally, the system was tested against the KDDCUP99 dataset and achieved increased overall accuracy compared with other state-of-the-art methods. Our approach achieves 99.2% accuracy on KDDCUP99. Future work should focus on strengthening the system to reduce false alarms.

Author Contributions: Conceptualization, O.M.E.; Formal analysis, H.A. and O.M.E.; Methodology, H.A. and O.M.E.; Project administration, S.E.; Resources, H.A.; Software, H.A.; Supervision, O.M.E. and S.E.; Validation, S.E.; Writing—review & editing, H.A., O.M.E. and S.E. All authors have read and agreed to the published version of the manuscript.

Funding: The authors did not receive any funding specifically for this work.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset is available online at https://datahub.io/machine-learning/kddcup99 (Last Accessed on 23 June 2022).

Conflicts of Interest: The authors state that they have no conflicting interests to disclose in connection with this work.

References

1. Gavalas, D.; Nicopolitidis, P.; Kameas, A.; Goumopoulos, C.; Bellavista, P.; Lambrinos, L.; Guo, B. Smart Cities: Recent Trends, Methodologies, and Applications. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 7090963. https://doi.org/10.1155/2017/7090963.
2. Elzeki, O.; Sarhan, S.; Abdelfatah, M.; Salem, H.; Shams, M. Biomedical Healthcare System for Orthopedic Patients Based on Machine Learning. *J. Eng. Appl. Sci.* **2021**, *16*, 616–622.
3. Anthopoulos, L.G. The Rise of the Smart City. In *Understanding Smart Cities: A Tool for Smart Government or an Industrial Trick*? Springer: Cham, Switzerland, 2017; pp. 5–45.
4. ITU-T FG-SCC: Setting the framework for an ICT architecture of a smart sustainable city. Focus Group Technical Specifications. Available online: http://www.itu.int/en/ITU-T/focusgroups/ssc/ Documents/website/web-fg-ssc-0345-r5-ssc_architecture.docx. (accessed on 10 March 2022).
5. Nagothu, D.; Xu, R.; Nikouei, S.Y.; Chen, Y. A Microservice-enabled Architecture for Smart Surveillance using Blockchain Technology. In Proceedings of the 2018 IEEE International Smart Cities Conference, ISC2 2018, Kansas City, MO, USA, 16–19 September 2018; pp. 1-4, doi:10.1109/ISC2.2018.8656968.
6. Gao, J.; Chai, S.; Zhang, B.; Xia, Y. Research on Network Intrusion Detection Based on Incremental Extreme Learning Machine and Adaptive Principal Component Analysis. *Energies* **2019**, *12*, 1223. https://doi.org/10.3390/en12071223.
7. Markit IH. The Internet of Things: a movement, not a market. *IHS Market.* **2017**, *1*, 1.
8. Steinberg, J. *Official (ISC)2 Guide to the CISSP-ISSMP CBK*; CISSP: Clearwater, FL, USA, 2015.
9. El-Hasnony, I.M.; Elzeki, O.M.; Alshehri, A.; Salem, H. Multi-Label Active Learning-Based Machine Learning Model for Heart Disease Prediction. *Sensors* **2022**, *22*, 1184. https://doi.org/10.3390/s22031184.
10. Jiang, S.; Song, X.; Wang, H.; Han, J.-J.; Li, Q.-H. A clustering-based method for unsupervised intrusion detections. *Pattern Recognit. Lett.* **2006**, *27*, 802–810. https://doi.org/10.1016/j.patrec.2005.11.007.
11. Antonakakis, M.; April, T.; Bailey, M.; Bernhard, M.; Bursztein, E.; Cochran, J.; Durumeric, Z.; Halderman, J.A.; Invernizzi, L.; Kallitsis, M.; et al. Understanding the Mirai Botnet. In Proceedings of the 26th USENIX Security Symposium, Vancouver, BC, Canada, 16–18 August 2017.
12. Santos, J.; Leroux, P.; Wauters, T.; Volckaert, B.; De Turck, F. Anomaly Detection for Smart City Applications over 5G Low Power Wide Area Networks. In Proceedings of the IEEE/IFIP Network Operations and Management Symposium: Cognitive Management in a Cyber World, NOMS 2018, Taipei, Taiwan, 23–27 April 2018.
13. Zhang, L.; Wang, X.; Jiang, Y.; Yang, M.; Mak, T.; Singh, A.K. Effectiveness of HT-assisted sinkhole and blackhole denial of service attacks targeting mesh networks-on-chip. *J. Syst. Arch.* **2018**, *89*, 84–94. https://doi.org/10.1016/j.sysarc.2018.07.005.
14. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A Detailed Analysis of the KDD CUP 99 Data Set. In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, ON, Canada, 8–10 July 2009.
15. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep Learning Approach for Network Intrusion Detection in Software Defined Networking. In Proceedings of the 2016 International Conference on Wireless Networks and Mobile Communications, WINCOM 2016: Green Communications and Networking, Fez, Morocco, 26–29 October 2016.
16. Rawat, S.; Srinivasan, A.; Ravi, V.; Ghosh, U. Intrusion detection systems using classical machine learning techniques vs integrated unsupervised feature learning and deep neural network. *Internet Technol. Lett.* **2022**, *5*. https://doi.org/10.1002/itl2.232.
17. Wang, B.; Sun, Y.; Yuan, C.; Xu, X. LESLA: A Smart Solution for SDN-Enabled MMTC E-Health Monitoring System. In Proceedings of the 8th ACM MobiHoc 2018 Workshop on Pervasive Wireless Healthcare Workshop, Mo-bileHealth 2018, Los Angeles, CA, USA, 25–26 June 2018.
18. Dey, S.K.; Rahman, M.M.; Uddin, M.R. Detection of Flow Based Anomaly in Openflow Controller: Machine Learning Approach in Software Defined Networking. In Proceedings of the 4th International Conference on Electrical Engineering and Information and Communication Technology, iCEEiCT 2018, Dhaka, Bangladesh, 13–15 September 2018.
19. Latah, M.; Toker, L. Towards an efficient anomaly-based intrusion detection for software-defined networks. *IET Netw.* **2018**, *7*, 453–459. https://doi.org/10.1049/iet-net.2018.5080.

20. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep Recurrent Neural Network for Intrusion Detection in SDN-Based Networks. In Proceedings of the 2018 4th IEEE Conference on Network Softwarization and Workshops, NetSoft 2018, Montreal, QC, Canada, 25–29 June 2018.
21. Latah, M.; Toker, L. An efficient flow-based multi-level hybrid intrusion detection system for Software-Defined Networks. *CCF Trans. Netw.* **2020**, *3*, 261–271. https://doi.org/10.1007/s42045-020-00040-z.
22. Zheng, D.; Hong, Z.; Wang, N.; Chen, P. An Improved LDA-Based ELM Classification for Intrusion Detection Algorithm in IoT Application. *Sensors* **2020**, *20*, 1706. https://doi.org/10.3390/s20061706.
23. Al-Yaseen, W.L.; Othman, Z.A.; Nazri, M.Z.A. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Syst. Appl.* **2017**, *67*, 296–303. https://doi.org/10.1016/j.eswa.2016.09.041.
24. Rani, M.; Gagandeep Effective network intrusion detection by addressing class imbalance with deep neural networks multimedia tools and applications. *Multimed. Tools Appl.* **2022**, *81*, 8499–8518. https://doi.org/10.1007/s11042-021-11747-6.
25. Imrana, Y.; Xiang, Y.; Ali, L.; Abdul-Rauf, Z. A bidirectional LSTM deep learning approach for intrusion detection. *Expert Syst. Appl.* **2021**, *185*, 115524. https://doi.org/10.1016/j.eswa.2021.115524.
26. Chen, L.; Gao, S.; Liu, B. An improved density peaks clustering algorithm based on grid screening and mutual neighborhood degree for network anomaly detection. *Sci. Rep.* **2022**, *12*, 1409. https://doi.org/10.1038/s41598-021-02038-z.
27. Ramadan, R.A.; Emara, A.-H.; Al-Sarem, M.; Elhamahmy, M. Internet of Drones Intrusion Detection Using Deep Learning. *Electronics* **2021**, *10*, 2633. https://doi.org/10.3390/electronics10212633.
28. Chung, Y.Y.; Wahid, N. A hybrid network intrusion detection system using simplified swarm optimization (SSO). *Appl. Soft Comput.* **2012**, *12*, 3014–3022. https://doi.org/10.1016/j.asoc.2012.04.020.
29. Ambusaidi, M.A.; He, X.; Tan, Z.; Nanda, P.; Lu, L.F.; Nagar, U.T. A Novel Feature Selection Approach for Intrusion Detection Data Classification. In Proceedings of the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2014, Beijing, China, 24–26 September 2014.
30. Khalvati, L.; Keshtgary, M.; Rikhtegar, N. Intrusion Detection based on a Novel Hybrid Learning Approach. *J. AI Data Min.* **2018**, *6*, 157–162. https://doi.org/10.22044/JADM.2017.979.
31. Mohammadi, S.; Mirvaziri, H.; Ghazizadeh-Ahsaee, M.; Karimipour, H. Cyber intrusion detection by combined feature selection algorithm. *J. Inf. Secur. Appl.* **2018**, *44*, 80–88. https://doi.org/10.1016/j.jisa.2018.11.007.
32. Alazzam, H.; Sharieh, A.; Sabri, K.E. A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer. *Expert Syst. Appl.* **2020**, *148*, 113249. https://doi.org/10.1016/j.eswa.2020.113249.
33. Jo, J.H.; Sharma, P.K.; Sicato, J.C.S.; Park, J.H. Emerging Technologies for Sustainable Smart City Network Security: Issues, Challenges, and Countermeasures. *J. Inf. Process. Syst.* **2019**. https://doi.org/10.3745/JIPS.03.0124.
34. Xu, J.; Palanisamy, B.; Ludwig, H.; Wang, Q. Zenith: Utility-Aware Resource Allocation for Edge Computing. In Proceedings of the 2017 IEEE 1st International Conference on Edge Computing, EDGE 2017, Honolulu, HI, USA, 25–30 June 2017.
35. Arasteh, H.; Hosseinnezhad, V.; Loia, V.; Tommasetti, A.; Troisi, O.; Shafie-Khah, M.; Siano, P. Iot-Based Smart Cities: A Survey. In Proceedings of the EEEIC 2016-International Conference on Environment and Electrical Engineering, Florence, Italy, 7–10 June 2016.
36. Mohanty, S.P.; Choppali, U.; Kougianos, E. Everything you wanted to know about smart cities: The Internet of things is the backbone. *IEEE Consum. Electron. Mag.* **2016**, *5*, 60–70. https://doi.org/10.1109/mce.2016.2556879.
37. Rahman, A.; Asyhari, A.T.; Leong, L.; Satrya, G.; Tao, M.H.; Zolkipli, M. Scalable machine learning-based intrusion detection system for IoT-enabled smart cities. *Sustain. Cities Soc.* **2020**, *61*, 102324. https://doi.org/10.1016/j.scs.2020.102324.
38. M., N. A Comprehensive Overview of Clustering Algorithms in Pattern Recognition. *IOSR J. Comput. Eng.* **2012**, *4*, 23–30. https://doi.org/10.9790/0661-0462330.
39. Salem, H.; El-Hasnony, I.M.; Kabeel, A.; El-Said, E.M.; Elzeki, O.M. Deep Learning model and Classification Explainability of Renewable energy-driven Membrane Desalination System using Evaporative Cooler. *Alex. Eng. J.* **2022**, *61*, 10007–10024. https://doi.org/10.1016/j.aej.2022.03.050.
40. Caruana, R.; Niculescu-Mizil, A. An Empirical Comparison of Supervised Learning Algorithms. In Proceedings of the ACM International Conference Proceeding Series, Santa Barbara, CA, USA, 23–27 October 2006.
41. Johnson, J.M.; Khoshgoftaar, T.M. Survey on deep learning with class imbalance. *J. Big Data* **2019**, *6*, 27. https://doi.org/10.1186/s40537-019-0192-5.
42. Liu, T.; Qi, A.; Hou, Y.; Chang, X. Method for Network Anomaly Detection Based on Bayesian Statistical Model with Time Slicing. In Proceedings of the World Congress on Intelligent Control and Automation (WCICA), Chongqing, China, 25–27 June 2008.
43. Vapnik, V.N. The Nature of Statistical Learning Theory. *Technometrics* **1997**, *38*, 409–409.
44. Kabir, E.; Hu, J.; Wang, H.; Zhuo, G. A novel statistical technique for intrusion detection systems. *Futur. Gener. Comput. Syst.* **2018**, *79*, 303–318. https://doi.org/10.1016/j.future.2017.01.029.
45. Fernandes, G.; Rodrigues, J.J.; Carvalho, L.F.; Al-Muhtadi, J.F.; Proença, M.L. A comprehensive survey on network anomaly detection. *Telecommun. Syst.* **2019**, *70*, 447–489. https://doi.org/10.1007/s11235-018-0475-8.
46. Brown, J.; Anwar, M.; Dozier, G. An Evolutionary General Regression Neural Network Classifier for Intrusion Detection. In Proceedings of the 2016 25th International Conference on Computer Communications and Networks, ICCCN 2016, Waikoloa, HI, USA, 1–4 August 2016.

47. Aburomman, A.; Reaz, M.B.I. A novel SVM-kNN-PSO ensemble method for intrusion detection system. *Appl. Soft Comput.* **2016**, *38*, 360–372. https://doi.org/10.1016/j.asoc.2015.10.011.
48. Bukhtoyarov, V.; Zhukov, V. Ensemble-Distributed Approach in Classification Problem Solution for Intrusion Detection Systems. In *International Conference on Intelligent Data Engineering and Automated Learning, Proceedings of the Intelligent Data Engineering and Automated Learning–IDEAL 2014, 15th International Conference, Salamanca, Spain, 10–12 September 2014*; Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Cham, Switzerland, 2014.
49. Safavian, S.; Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man, Cybern.* **1991**, *21*, 660–674. https://doi.org/10.1109/21.97458.
50. Kowsari, K.; Meimandi, J.K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text Classification Algorithms: A Survey. Information, Switzerland. *Information* **2019**, *10*, 150. https://doi.org/10.3390/info10040150.
51. Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. https://doi.org/10.1016/j.neucom.2005.12.126.
52. Zhang, K.; Hu, Z.; Zhan, Y.; Wang, X.; Guo, K. A Smart Grid AMI Intrusion Detection Strategy Based on Extreme Learning Machine. *Energies* **2020**, *13*, 4907. https://doi.org/10.3390/en13184907.
53. Pradhan, A.K.; Das, K.; Mishra, D.; Mishra, S. Exploration of Hyperparameter in Extreme Learning Machine for Brain MRI Datasets. In *Intelligent and Cloud Computing*; Smart Innovation, Systems and Technologies; Springer: Singapore, 2021.
54. Hafiz, F.; Swain, A.; Naik, C.; Abecrombie, S.; Eaton, A. Identification of power quality events: Selection of optimum base wavelet and machine learning algorithm. *IET Sci. Meas. Technol.* **2019**, *13*, 260–271. https://doi.org/10.1049/iet-smt.2018.5044.
55. UCI Machine Learning Repository KDD Cup 1999 Data. 1999. Available online: http//kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 15 February 2022).
56. Thomas, C.; Balakrishnan, N. Performance Enhancement of Intrusion Detection Systems Using Advances in Sensor Fusion. In Proceedings of the 11th International Conference on Information Fusion, FUSION 2008, Cologne, Germany, 30 June–3 July 2008.
57. Mahmoud, A.; Shams, M.Y.; Elzeki, O.M.; Awad, N.A. Using Semantic Web Technologies to Improve the Extract Transform Load Model. *Comput. Mater. Contin.* **2021**, *68*, 2711–2726. https://doi.org/10.32604/cmc.2021.015293.
58. Kumar, D.A.; Venugopalan, S.R. A Novel Algorithm for Network Anomaly Detection Using Adaptive Machine Learning. In *Progress in Advanced Computing and Intelligent Engineering*; Advances in Intelligent Systems and Computing; Springer: Singapore, 2018.
59. Salem, H.; Negm, K.R.; Shams, M.Y.; Elzeki, O.M. Recognition of Ocular Disease Based Optimized VGG-Net Models. In *Medical Informatics and Bioimaging Using Artificial Intelligence*; Studies in Computational Intelligence; Springer: Cham, Switzerland, 2016.
60. Salem, H.; Attiya, G.; El-Fishawy, N. Intelligent decision support system for breast cancer diagnosis by gene expression profiles. In Proceedings of the National Radio Science Conference, NRSC, Alexandria, Egypt, 23–25 February 2016.
61. Shams, M.Y.; Elzeki, O.M.; Elaraby, M.E.; Hikal, N.A. Signature Recognition Based on Support Vector Machine and Deep Convolutional Neural Networks for Multi-Region of Interest. *J. Theor. Appl. Inf. Technol.* **2020**, *98*, 3887–3897.