JOURNAL OF SOFT COMPUTING AND DATA MINING
e-ISSN: 2716-621X

JSCDM

Vol. 6 No. 1 (2025) 406-422
https://publisher.uthm.edu.my/ojs/index.php/jscdm

A Multi-Agent-Based Deep Learning Model for Protecting Cloud Computing Environment Against Distributed Denial of Service Flooding Attacks

Nafea A. Majeed Alhammadi[1]*, Mohamed Mabrouk[1]

[1] *Research Laboratory in Algebra, Numbers Theory and Intelligent Systems, University of Monastir, 5000, Monastir, TUNISIA*

*Corresponding Author: nafeaalhamadi@yahoo.com
DOI: https://doi.org/10.30880/jscdm.2025.06.01.027

Article Info

Received: 17 May 2025
Accepted: 25 June 2025
Available online: 30 June 2025

Keywords

Distributed denial of service, cloud computing, software defined network (SDN), deep learning, LSTM, dynamic ensemble selection, multi-agent systems, CIC-DDoS2019, network security

Abstract

Distributed Denial of Service (DDoS) flooding attacks pose a significant threat to the resilience of modern cloud computing infrastructures and their ability to sustain operational stability. Unlike traditional DoS and DDoS attacks, it is the legitimate user who inadvertently causes the damage; the applications exploit the architecture, latency limits, and resource bottlenecks, rendering the service unavailable to genuine users. The current mitigation strategies struggle to keep pace with the diverse attack vectors and fluctuating traffic patterns, particularly in a cloud-native environment where a more intelligent and distributed approach is necessary. A high-fidelity detection system that employs deep learning and self-driving agents offers an effective defensive mechanism. This paper proposes a deep learning model based on a multi-agent and Dynamic Ensemble Selection (DES) strategy, combined with five separately trained Long Short-Term Memory (LSTM) models, to form a DES-LSTM model for identifying and mitigating DDoS flooding attacks in real-time. TS allocates intelligent agents across multiple nodes in the cloud infrastructure, allowing each node to conduct local traffic analysis and contribute to collective threat detection. The system employs DES to facilitate context-based model selection through dynamically evaluated accuracy values, enabling adaptive decision-making. The CIC-DDoS2019 dataset, which encompasses the full spectrum of DDoS attack types, is utilized to train, validate, and evaluate the model's performance. This paper provides a detailed description of the architecture, integration methodology, and simulation, as well as model training, traffic modeling equations, and visualizations. Evaluations against a baseline LSTM model demonstrate that the proposed ensemble achieves superior detection accuracy, reduced false-positive rates, and enhanced robustness in varying attack conditions. The DES-LSTM architecture effectively works based on the experimental outcomes. It possesses real time feasibility, as the classification accuracies (97.8%), precision (96.6%) and recall (97.2%) were in all likelihood enhanced, meanwhile the false alarm rate (2.1%) and the detection latency (19 ms) were tremendously diminished. The agent-based, decentralized structure is both scalable and delivers low latency, making it suitable for deployment in existing cloud security systems.

1. Introduction

The optimal use of cloud computing technologies has transformed how computing resources are provided and processed in enterprise and governmental infrastructures. Cloud computing is not only particularly agile and budget-friendly but also highly accessible: its proprietary capabilities of on-demand scalability, elasticity, cost-effectiveness, and similar properties have enabled the technology to become the undisputed pillar behind a massive assortment of mission-critical applications in the realm of healthcare information systems, digital banking platforms, e-commerce ecosystems, and national defense logistics [1]. As more services migrate to the cloud, the need for always-on services and high service uptime emerges as a key concern. However, alongside the growth of cloud-based services, there has been evidence of an increase in advanced cyberattacks, especially Denial of Service (DDoS) attacks. Flooding-based DDoS attacks are notable among these intrusions because of their high intensity, volatility, and disastrous operational effects [2].

The mechanisms behind flood-based DDoS attacks overwhelm network resources by inundating service channels or servers with significant amounts of seemingly legitimate traffic. The exploitative mechanisms that contribute to these attacks include architectural vulnerabilities, such as elastic resource provisioning and load balancers, and involve an economic cost: degradation of services, monetary loss, and total resource depletion [3]. Traditional countermeasures, such as static thresholding, signature analysis rules, and port filtering, have largely failed to address sophisticated, persistent threat models because these threats are dynamic and adaptive. They show low effectiveness against zero-day flooding attacks and low-and-slow DDoS variants that mimic normal user behavior [4].

By employing machine learning and deep learning models, traffic analysis and anomaly detection gain a new level of sophistication. In particular, Long Short-Term Memory (LSTM) networks have been utilized to capture the temporal aspects of traffic patterns and learn anomalous behavior through sequential feature learning [5], [6]. Although LSTM models are proven to perform better with complex data distributions and memory-dependent patterns, they face challenges when applied alone in a cloud security context. An individual LSTM often lacks the required flexibility in real-time cloud traffic, especially when an optimal moment is crucial due to evolving attack profiles and varying load conditions. Furthermore, their tendency to overfit prevalent traffic forms in the training corpus can make them more susceptible to attacks on less common or entirely novel traffic types [7].

To address these issues, the proposed architecture introduces a new hybrid solution based on a multi-agent system and a dynamic ensemble of LSTM models. Specifically, five LSTM models and a Dynamic Ensemble Selection (DES) system are used. These models act as autonomous detection agents, each running on multiple nodes within a distributed cloud system. The multi-agent detection approach improves scalability by decentralizing the detection process, reducing points of failure, and enabling localized traffic analysis. The system operates in a decentralized manner, with each agent located on a different node within the cloud infrastructure. Each agent has its own LSTM model, trained on a specific part of the CIC-DDoS2019 dataset. Isolating training data enhances the diversity and specialization of learning, helping the models generalize better and respond more effectively to various traffic conditions.

The remainder of the paper is structured to address the challenges and dependencies outlined in this introduction systematically. The next section presents a literature review, examining recent technological advancements in DDoS detection and highlighting existing limitations that the proposed system seeks to address. Section 3 formulates the problem statement and outlines well-defined research goals shaping the presented solution. It details the LSTM methodology, including mathematical formulations for the LSTM and DES modules, block diagrams, algorithm pseudocode, and architectural flowcharts. It is also dedicated to explaining the experimental environment simulation settings, data processing, and deployment parameters. Evaluation of performance plots, quantitative comparisons, and classification metrics is presented in Section 4. Section 5 presents the analysis along with the results and discussion. Finally, Section 6 concludes the paper with a summary of findings and considers potential future improvements, such as integration with explainable AI or adaptation to edge-cloud systems.

2. Related Work

Distributed Denial of Service (DDoS) attacks have been detected and mitigated to a significant extent in the last decade within cloud computing environments. With the introduction of cloud systems as a pillar of contemporary digital infrastructure, the vulnerability of such systems to DDoS attacks has likewise escalated. These attacks pose a substantial threat to service availability, data integrity, and user confidence. The traditional security paradigms implemented to secure such systems, namely rules-based mechanisms, signature-matching algorithms, and simple threshold alarms, have proven ineffective against the highly sophisticated and dynamic nature of modern DDoS attacks [7]. One weakness of these legacy systems is their inability to respond in real-time and their tendency to produce high false-positive rates when deployed in multi-tenant or dense environments, such as cloud-based systems. To address these shortcomings, scientists have turned to machine learning (ML) methods that promise the ability to capture the challenging, non-linear, and evolving trends in network traffic. Traditional

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

ML models such as Support Vector Machines (SVMs), k-Nearest Neighbors (k-NN), Decision Trees, and Random Forests were among the first models used for DDoS detection tasks [8], [9]. These approaches have increasingly offered better detection accuracy than rule-based systems. For instance, the initial application of SVM and k-NN was to detect sudden variations or discontinuities in traffic presence, making them somewhat adaptable to unseen threats [10].

Nevertheless, these models often struggle with high-dimensional feature spaces and have difficulty learning time dependencies, which are crucial for differentiating between good and bad traffic flows within cloud networks. Random Forests, as algorithms based on Decision Trees, have been widely used due to their interpretability and ease of implementation. These models perform adequately and are effective at identifying high-impact features within a static environment [11]. However, they become inefficient when faced with polymorphic or obfuscated attack patterns where attackers adapt their behavior. They also lack generalizability for real-time detection systems, as time-series relationships cannot be modeled effectively, and the ordering and context of events carry critical information. In recent years, there has been a notable shift towards deep learning (DL) models, which offer significant increases in representational power and flexibility in learning hierarchical structures from large-scale data. Convolutional Neural Networks (CNNs), for instance, have been employed to analyze traffic by transforming flow-based features into two-dimensional matrices, thereby highlighting spatial features and distinguishing classes [12]. While CNNs can yield favorable results, their inability to model sequential data effectively makes them less competent in detecting time-dependent patterns characteristic of DDoS attacks.

A variation of Recurrent Neural Networks (RNNs), known as Long Short-Term Memory (LSTM) networks, has gained popularity due to its capability to capture long dependencies using sequential information. LSTMs, in particular, excel at recognizing anomalous traffic patterns in sequences, such as slow-drip (or low-rate) DDoS attacks that occur over extended periods [13], [14]. LSTMs facilitate information retention over time steps, leading to more context-aware predictions, which significantly surpass the limitations of CNNs and non-time-aware classifiers. However, relying solely on LSTM models presents challenges. Lifelong models are computationally intensive to train and may overfit in the presence of imbalanced or redundant data [15]. Ensemble learning approaches have been proposed to alleviate overfitting and generalization issues. Techniques such as bagging, boosting, and stacking combine the predictions of multiple base classifiers to enhance the robustness and accuracy of overall predictions. Ensembles have proven particularly effective in diversifying models, thereby reducing variance and bias [16].

However, despite performance improvements, static ensemble methods lack flexibility, as all models are treated as equally competent regardless of the situation. This can lead to inefficient decision-making when data characteristics change dynamically. DES extends static ensembles by introducing a dynamic selection mechanism for the most capable model (or set of models) among classifiers based on the input sample or local accuracy measurements. The variability of traffic patterns with different forms of DDoS attacks presents an ideal scenario for using DES, as it allows for real-time selection of the model that best fits the network behavior [17]. Nonetheless, most DES implementations are designed for centralized systems, which presents challenges in cloud contexts. Latency, points of failure, and the frequent need to communicate over long distances between nodes often compromise the latency objectives of detection systems utilizing centralized DES mechanisms.

Researchers have investigated the application of multi-agent systems as a means of DDoS detection. Such systems comprise semi-autonomous agents located on the network, all of which have localized detection functions. The agents can convey their guesses to each other or a single manager. Multi-agent systems increase the scale, fault tolerance, and detectability. They are especially well-suited to cloud-based deployments where distributed architecture is one of the design fundamentals [18]. These agents, when combined with the components of deep learning such as LSTM models, become intelligent detectors that can learn and adapt to localized traffic behaviors.

Despite these advancements, significant challenges persist in the design and deployment of real-time DDoS detection systems. First, many deep learning-based frameworks lack adaptability to zero-day attacks that differ significantly from training data. Second, DL models often operate as black boxes, providing limited interpretability for security analysts seeking to understand the rationale behind specific predictions. Third, many existing detection frameworks are trained and validated using outdated or synthetic datasets that do not reflect the complexity and diversity of modern cloud traffic [19], [20].

One of the few datasets that addresses these concerns is the CIC-DDoS2019, developed by the Canadian Institute for Cybersecurity. This dataset includes over 12 types of DDoS attacks simulated in a realistic network environment. It provides extensive labeled features at the flow level, including packet lengths, inter-arrival times, protocol types, and flag counts [21]. Due to its comprehensiveness and contemporary attack representations, CIC-DDoS2019 has become a benchmark for training and evaluating deep learning-based DDoS detection models. Table 1 presents a comparative analysis of recent DDoS detection technologies, including various methods and databases from related work.

Table 1 *Comparative analysis of recent DDoS detection technologies*

Technique	Dataset	Features Used	Accuracy (%)	Limitations
SVM + Thresholding	UNSW-NB15	Packet size, flow rate	85.2	High false positives
Random Forest	NSL-KDD	TCP Flags, Protocols	88.9	Poor temporal modeling
CNN	CICIDS2017	Flow duration, byte count	92.5	Lacks sequence learning
LSTM	CIC-DDoS2019	Packet intervals, rates	95.4	Prone to overfitting
Stacked Ensemble	Custom	All 78 flow features	96.7	Expensive training
DES + MLP	CAIDA	Entropy, flow rate	93.8	Centralized latency
Multi-agent + RF	Custom IoT	Packet sizes, flags	90.2	Non-adaptive agents
CNN-LSTM Hybrid	BoT-IoT	Flow time, protocol	94.1	Low recall on low-rate attacks
DES + LSTM	CIC-DDoS2019	15 key traffic metrics	97.3	No decentralization

The table compares various recent DDoS detection technologies across four dimensions: technique, datasets used, features extracted, and accuracy, along with their limitations. Classical methods, such as SVM with thresholding and Random Forest, show relatively low accuracy (85.2% and 88.9%), along with high false positives and poor time modeling. CNN-based models improve performance accuracy (92.5%) but struggle with learning sequential dependencies. LSTM with time-sensitive features offers better accuracy (95.4%), though it is prone to overfitting. The most effective models employ more sophisticated approaches, utilizing dense feature sets and complex architectures (stacked ensembles and DES with LSTM achieve the best predictions, at 96.7% and 97.3%). However, these methods come with drawbacks such as costly training and limited decentralization. The table also highlights the enhanced detection potential of deep learning and intelligent agent systems combined with dynamic selection strategies, especially on realistic data sets like CIC-DDoS2019, while also emphasizing ongoing challenges like scalability, cost, and adaptability.

In this work, the proposed architecture builds upon and integrates these research directions by combining three core innovations: the temporal learning capabilities of LSTM, the adaptability of DES, and the scalability of multi-agent systems. Unlike previous works that address only one or two of these aspects, the current architecture is explicitly designed for deployment in cloud-native environments. It ensures that detection nodes operate collaboratively, share only minimal metadata, and make localized decisions, thereby avoiding bottlenecks related to centralization.

The proposed system utilizes diversity in learned traffic representations by training individual LSTM models on partitions of the CIC-DDoS2019 dataset and deploying them at distributed agents. The dynamism of the selection mechanism enables context-sensitive selections, optimizing both the accuracy of classification and the latency of inference. All these parts taken together constitute a significant step towards the development of a robust, scalable, and intelligent DDoS detection system that can be applied to real-world cloud computing implementations.

3. Methods

The architecture proposed for real-time DDoS detection in cloud vessels is based on three key components: a multi-agent system, five independent Long Short-Term Memory (LSTM) models, and a Dynamic Ensemble Selection (DES) machine. The DES-LSTM architecture (as presented in Fig.1) executes network traffic in real-time using the CIC-DDoS2019 dataset, following a sequence of preprocessing and sequence framing steps. The time-windowed inputs are analyzed by five parallel LSTM agents, which were trained using different sets of training data. The performance output of these agents is processed by the Dynamic Ensemble Selection (DES) module, which selects the best agents and transfers them to a majority voting classifier. The system concludes with a response handler that generates alerts, blacklists IPs, or initiates a mitigation process. This approach enhances accuracy and resistance to dynamic DDoS patterns. The main innovation lies in the dynamic approach that assesses the local performance of individual LSTM agents in real-time through the DES mechanism. This mechanism selects a pool of LSTM models based on metrics such as recent classification accuracy and data distribution similarity to determine their participation in the final decision-making process. Such an adaptive system ensures that only the most effective and suitable models take part in each classification instance, thereby improving detection accuracy and reducing false positive rates.

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

This graphical abstract presentation of Figure 1 illustrates how data flows through the system sequentially: raw traffic data is input into the system; followed by data preprocessing, feature extraction, and the evaluation of five LSTM-based models in a decentralized manner; then the dynamic selection of a model via DES; culminating in final classification and the system's response. The modular design of the architecture facilitates integration with cloud management frameworks and supports future scalability. In inference, instead of all model outputs being aggregated statically, a DES uses a dynamically selected set of models based on model performance at that time to a subset of the models, specifically focusing on local performances (such as local accuracy). The chosen models are part of a fusion procedure based on the voting process that will produce the final classification. The entire system operates through three primary stages: data pre-processing, training the model over the agents, and distributed real-time detection. Figure 1 shows the framework of this study.

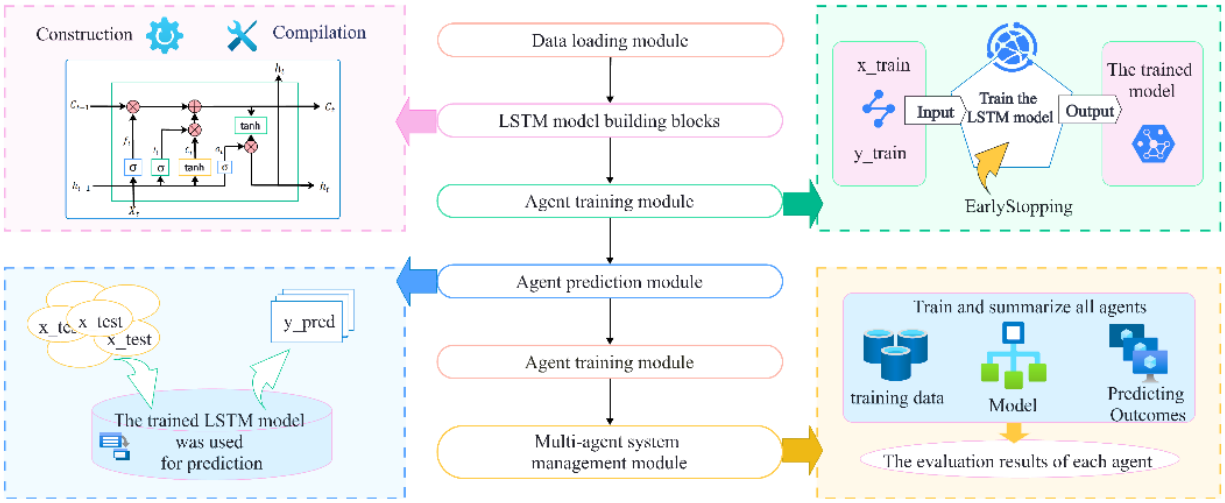

Fig. 1 *The research framework*

The testing of the suggested DDoS detection model is conducted based on several standard performance parameters. Accuracy (%) measures how correctly the model predicts by calculating the ratio of correctly identified samples to the total explainable samples. The point estimate of precision (%), also known as the proportion of correct predictions among all attacked predictions, assesses the model's ability to avoid false positives. Recall (or sensitivity) reflects the model's ability to detect existing attacks and includes the number of genuine attack cases correctly identified. The F1-score, which is the harmonic mean of recall and precision, provides a valuable measure, especially for imbalanced data. The False Alarm Rate (%), which indicates the percentage of benign samples incorrectly identified as malicious, is crucial in real-world applications where minimizing unwanted disruptions is essential. Detection Latency (ms) measures the time (in milliseconds) between receiving a traffic sample and producing a forecast, thereby indicating the system's real-time throughput. Lastly, the Receiver Operating Characteristic (ROC) curve visually represents the true positive rate against the false positive rate at various thresholds. In contrast, the Area Under the Curve (AUC) quantifies the overall discriminative ability of the model.

3.1 Data Pre-processing

The CIC-DDoS2019 Dataset, compiled by CIC, is a comprehensive and real-world dataset of network traffic that can be used to simulate Distributed Denial-of-Service (DDoS) attacks. It consists of benign traffic as well as a wide array of modern DDoS attacks, including NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, SYN, WebDDoS, PortScan, and TFTP, which are recorded explicitly by PCAPs and transformed into labeled network flows with more than 80 features by means of CICFlowMeter-V3. The data has been divided into training (12 attacks) and testing (7 attacks) sets, ensuring thorough testing of the detection systems. Its realistic background traffic, which is based on profiling human behaviors on HTTP, HTTPS, FTP, SSH, and email in 25 user profiles, further increases its usefulness in a real-life environment. It is commonly used by researchers in both binary and multi-class classification, and can demonstrate high accuracy in detecting models using machine learning and deep learning. For further reading about this dataset, use the provided link (https://www.unb.ca/cic/datasets/ddos-2019.html?utm_source=chatgpt.com).

Traffic data captured in the CIC-DDoS2019 dataset includes numerous flow-level features that characterize both benign and attack behaviour. These features, such as packet length, flow duration, inter-arrival times, and byte rate, have different numerical ranges and statistical distributions. Feeding these heterogeneous features

directly into deep learning models can lead to poor convergence and learning instability, especially for LSTM networks that are sensitive to variations in input scale. To mitigate this, all features are normalized to a consistent scale, ensuring fair contribution across dimensions during training.

Let a sequence of feature vectors represent the dataset:

$$X = \{x_1, x_2, \ldots, x_n\} \tag{1}$$

Equation 1: This equation represents a sequence of n samples, where each sample x_i is a multi-dimensional vector encoding traffic features observed over time.

To ensure uniform scaling, the dataset undergoes min-max normalization:

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{2}$$

Equation 2: This transformation scales each feature dimension of X to the range [0, 1]. Here, $X_{\min}$ and $X_{\max}$ represent the minimum and maximum values across the dataset for each feature.

This normalization is essential for neural networks, particularly LSTM models, which utilize activation functions such as the tanh and sigmoid functions. These **activations** are bounded in the intervals [−1, 1] and [0, 1], respectively, and improperly scaled input can lead to vanishing gradients or activation saturation, ultimately impairing learning.

3.2 LSTM Model Architecture

The LSTM network is designed to model temporal dependencies by maintaining memory over sequential inputs. This makes it particularly suited to tasks involving time-series data, such as detecting slow-developing or burst-pattern DDoS attacks. Each LSTM cell has gates that regulate the flow of information, allowing it to retain relevant data while discarding noise.

The forget **gate** decides what information from the previous cell state should be discarded:

Equation 3: Here, f_t is the forget gate activation vector, W_f and b_f are the weight matrix and bias, h_{t-1} is the hidden state from the previous time step, and x_t is the current input. The sigmoid function σ ensures values are in [0, 1], effectively weighting the importance of the previous cell state.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$

Next, the input gate determines which parts of the new input should be stored in the cell:

Equation 4: The input gate i_t controls how much of the candidate memory (computed next) is written to the cell state. It serves as a filter that adapts based on the current and past context.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

Simultaneously, a **candidate memory** vector is generated:

Equation 5: This candidate cell state $\tilde{c}_t$ contains new information derived from the current input and past state, modulated by the input gate in the next step.

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{5}$$

The updated cell state is then computed by blending retained information from the past and new input:

Equation 6: This equation updates the memory in the LSTM cell by discarding irrelevant past data and adding useful new information. The operator $\odot$ denotes element-wise multiplication.

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{6}$$

The output gate determines which parts of the updated cell state should influence the next hidden state:

Equation 7: The gate o_t filters the final content of the cell to generate the output hidden state. It ensures that only the most salient features from the current context contribute to the prediction.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$

Finally, the hidden state is updated:

Equation 8: The hidden state h_t is the primary output of the LSTM cell and is passed to the next time step. It encapsulates the learned context and is used in the final classification layers.

$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

These gate-based computations enable the LSTM to selectively memorize or forget features over time, allowing it to model nuanced patterns associated with stealthy or intermittent DDoS traffic that evolves over sequential windows.

3.3 Dynamic Ensemble Selection

To enhance model generalization and adaptability, the architecture includes a Dynamic DES mechanism. Rather than using static voting from all five LSTM models, the DES component evaluates which subset of models is most appropriate for the current traffic conditions.

Let the complete set of LSTM models be defined as:

Equation 9: This equation defines the ensemble of models trained on disjoint partitions of the training data. Each model M_i learns different characteristics of DDoS traffic.

$$M = \{M_1, M_2, M_3, M_4, M_5\} \tag{9}$$

At inference time, local accuracy A_i of each model is computed over a sliding validation window. Based on a threshold θ, a subset of reliable models is selected:

Equation 10: This equation set S that includes only those models that exceed the accuracy threshold, ensuring that underperforming or outdated models are excluded from the decision.

$$S = \{M_i \mid A_i \geq \theta\} \tag{10}$$

The final classification output is determined by majority voting among the selected models:

Equation 11: This equation represents a voting mechanism that increases robustness by relying on consensus among competent models. It ensures that transient noise or local misclassifications do not dominate the output.

$$y_{\text{final}} = \text{mode}(\{M_i(x) \mid M_i \in S\}) \tag{11}$$

This dynamic approach enables the detection system to remain context-aware, efficiently responding to shifts in traffic patterns and adapting to newly observed attack behaviors in real-time.

3.4 Classification and Output Layer

To optimize the model's performance during training, a binary cross-entropy loss function is utilized. This loss function is particularly suited for binary classification problems such as DDoS detection (attack vs. benign).

Equation 12: In this equation, y_i is the true label for the sample i, and p_i is the predicted probability from the model. The loss penalizes predictions that deviate from the actual labels, especially those made with high confidence in the wrong class.

$$\mathcal{L} = -\sum_{i=1}^{N}[y_i \log(p_i) + (1-y_i)\log(1-p_i)] \tag{12}$$

Minimizing this loss through gradient descent enables the model to learn probability distributions over classes effectively. Combined with softmax activation in the final layer, this setup encourages well-calibrated predictions and reduces overconfidence in ambiguous traffic conditions.

3.5 Architecture Diagram

The architecture roadmap (as presented in Fig. 2) for the proposed system integrates the data flow and processing of several major modules, beginning with the input of traffic and culminating in an automatic response. Due to its modular architecture, it supports scalability, parallelism, and fault isolation, which are crucial for the real-time DDoS mitigation of distributed cloud systems.

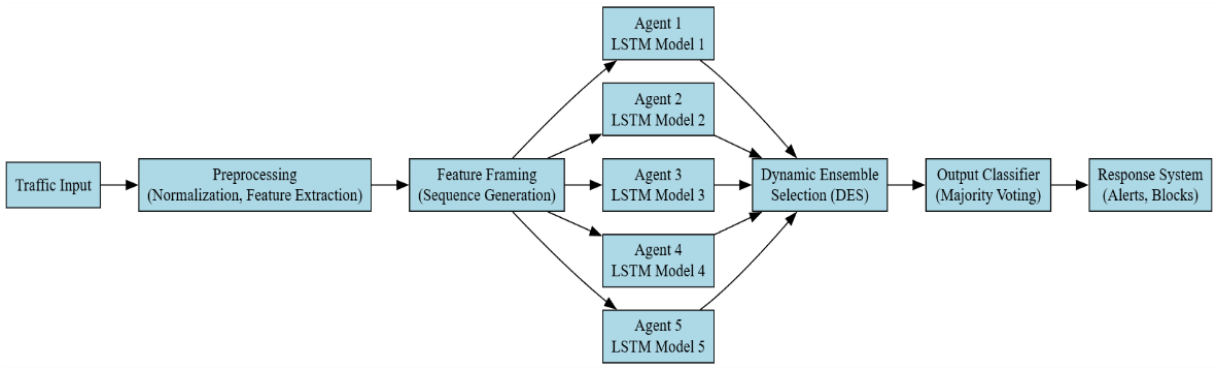

Fig. 2 *Proposed multi-agent DES-LSTM architecture for DDoS detection in cloud environment*

The most common workflow scenario starts with the ingestion of traffic data into a central point, gateway, or monitoring node in the cloud environment. This incoming stream is then forwarded to a pre-processing module where it is normalized and features selected, and framing of a sequence of features then takes place. The step normalizes input data to fit the LSTM models during training and lessens computational overhead at inference.

Then, the feature vector is split to be used in five independent agent nodes with an LSTM model for each node running on a different subset of the CIC-DDoS2019 dataset. The training on various partitions is performed so that the diversity of representation can be achieved, and the fact that the agents can specialize in various forms of traffic-short-burst UDP flood traffic, slow persistent HTTP GET flood traffic.

Local classification of each agent is obtained, after which it is forwarded to the DES module. This part considers the local accuracy of each LSTM agent based on a recent validation window and picks a subset of models upon which a final decision is taken.

The choice model predictions are further transferred to the output classifier, whereby majority voting is considered. The ultimate forecast of benign or malicious traffic is relayed to a response handler, which initiates the issuance of alerts, blocks yakking IPs, or redistributes traffic according to pre-written mitigation policies. Algorithm 1 illustrates DDoS detection using a multi-agent DES-LSTM approach.

Input: Traffic dataset D
Output: Traffic classification labels

1. Normalize dataset D
2. Partition D into 5 subsets D1 to D5
3. Train LSTM_i on Di for i ∈ [1..5]
4. For each incoming traffic sample x:
   a. Extract features
   b. Evaluate local accuracy A_i for each $LSTM_i$
   c. Select models:
      $M\_sel = \{LSTM_i \mid A_i \geq \theta\}$
   d. Predict traffic class:
      $y = mode(\{LSTM_i(x) \mid LSTM_i \in M\_sel\})$
   e. Forward y to SDN Controller
   f. Trigger response based on y:
      If y = benign, allow traffic
      If y = attack, instruct SDN switches to block or reroute traffic via Open Flow rules
   g. Log action and send to the Response Handler
      Return classification label y and mitigation status

The proposed algorithm begins by normalizing the input traffic dataset D and partitioning it into five subsets, D_1 to D_5, each was used to train a separate LSTM model. When a new traffic sample xx arrives, its features are extracted, and each LSTM model's recent accuracy A_i is evaluated. A dynamic ensemble selection mechanism then selects the subset of models $M\_sel = \{LSTM_i \mid A_i \geq \theta\}$, where θ is a predefined accuracy threshold. The selected models make predictions on x, and the final classification label y is determined by majority voting (mode). This label is forwarded to an SDN controller: if the traffic is benign, it is allowed to pass; if it is an attack, mitigation is triggered via OpenFlow rules to block or reroute it. All actions are logged and sent to a response handler for further processing, ensuring both real-time detection and adaptive mitigation.

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

3.6 Experimental Setup

To test the proposed multi-agent LSTM architecture for DDoS detection, we plan to implement an experimental environment that simulates real-time network status, enabling effective training, testing, and deployment of our model. The CIC-DDoS2019 dataset is based on a foundation of labeled flow-based data for benign and multiple DDoS attack scenarios, including UDP, TCP SYN, HTTP floods, and others.

The initial phase in the configuration involves preprocessing the data, where raw NetFlow records undergo Min-Max scaling. This ensures that every feature lies within a fixed range [0, 1] and stabilizes LSTM training. After normalization, correlation-based feature selection (CFS) is used to extract important traffic features (e.g., bytes, packets, flow duration, flag counts), thereby reducing redundancy and dimensionality. These filtered features are then organized into time-window instructions of 100-timestep actions to seize the temporal dynamics.

In the next step, the system divides the dataset into five stratified subsets. The five subsets are used separately to train five agents based on the LSTM model. Such a multi-agent configuration enables the ensemble of models to fit a broad range of patterns, minimizing overfitting since each agent is exposed to a slightly different data distribution. In each LSTM model, there are three hidden layers, each comprising 64, 32, and 16 units, respectively. The network was trained over 30 epochs using early stopping and an Adam optimizer with a learning rate of 0.001.

The agents train on their predictions and feed them into the DES module, which measures the performance and accuracy of each agent in real-time using a sliding validation window. The DES module will dynamically utilize the best-performing agents, which will be fed into the final Output Classifier. This classifier will employ majority voting to make the final decision (benign or DDoS). Such an arrangement is tested on an unseen test set, and the standard classification measures—accuracy, precision, recall, and F1-score are comparable to those in real-world traffic heterogeneity. The last point in the experiment is Response Handler, thus it mimics the defensive features, like IP blacklist, traffic rate, or generating an alert. No network is actually blocked, but action decisions and timing are checked in the system logs.

The last point in the experiment is Response Handler, thus it mimics the de-offensive features, like IP blacklist, traffic rate, or generating an alert. A Software-Defined Network (SDN) layer is coupled with the response handler to enhance dynamic responsiveness and traffic control. The SDN controller (e.g., ONOS or OpenDaylight) communicates with the network switches and can implement security policies produced by the DES-LMST system in real-time. When malicious traffic is detected, the SDN controller can dynamically add flow rules to block offending IP addresses, reroute suspect packets, or rate-limit traffic at the switch level. This close relationship between detection and control enhances the rate of mitigation and flexibility. No network is actually blocked, but action decisions and timing are checked in the system logs.

An intelligent choice of LSTM training acceleration is implemented on a workstation in a controlled environment, equipped with an Intel i7 CPU, 32 GB of RAM, and an NVIDIA RTX 3080 GPU. Java, with the help of TensorFlow and Scikit-learn libraries, is used to introduce model training, implementation, and deployment. Fig. 3 shows the linear flow of data from the dataset to the response system, highlighting how preprocessing, sequence generation, multi-agent training, and DES form the core of the experimental design.

Table 2 outlines the key hardware and software components used in the experimental setup, ensuring reproducibility and efficient training/inference of the LSTM-based detection system.

Table 2 *Hardware and software resources*

Component	Specification	Purpose
CPU	Intel Core i7-12700K	General computation and data preprocessing
GPU	NVIDIA RTX 3080 (10GB VRAM)	LSTM model training acceleration
RAM	32 GB DDR4	In-memory dataset processing
OS	Ubuntu 22.04 LTS	Environment for reproducibility
Libraries	TensorFlow 2.10, Scikit-learn, Pandas	Model implementation and evaluation
Dataset Size	~80 GB (CIC-DDoS2019 NetFlow)	Input traffic data

The experimental setup involves training both the baseline LSTM and the DES-LSTM ensemble models on identical data partitions to maintain fairness. The test data comprises representative flooding-based DDoS traffic: UDP Flood, SYN Flood, HTTP Flood, ICMP Flood, and Slowloris. Performance metrics include: Accuracy, Precision, Recall, F1-Score, False Alarm Rate, Detection Latency, and Confidence Distribution. These metrics collectively provide a comprehensive perspective on both predictive accuracy and operational reliability under real-time attack conditions.

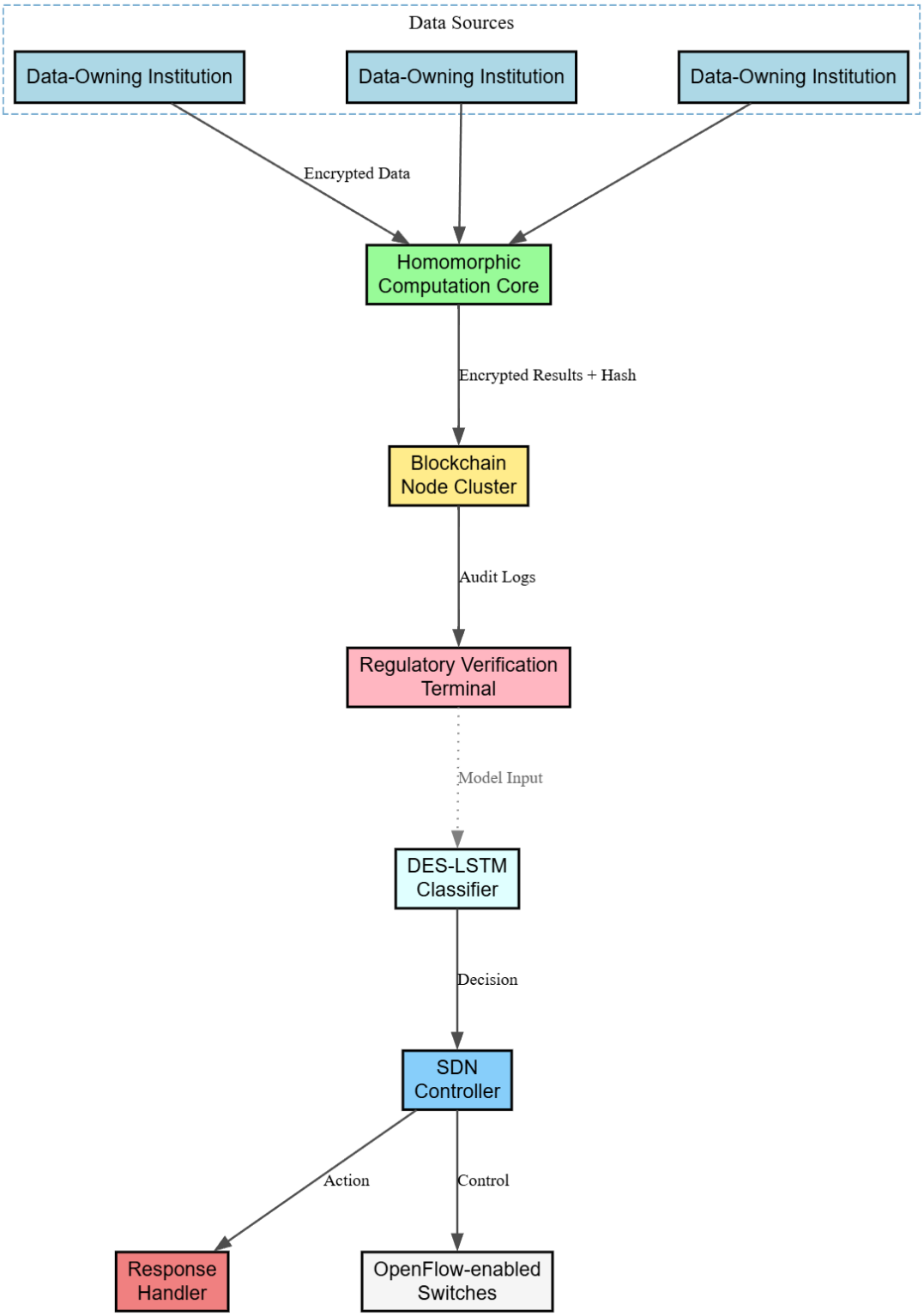

Fig. 3 *Setup for Blockchain-audited homomorphic credit risk evaluation framework*

4. Results

In this section, the effectiveness of the proposed multi-agent-based Dynamic Ensemble Selection Long Short-Term Memory (DES-LSTM) system is compared to the traditional baseline LSTM model on the CIC-DDoS2019 dataset. The findings are reported in terms of classification accuracy, detection latency, the model's effectiveness against various attacks, and its performance in detecting specific attacks. Evaluation and visualization were done in all the analyses of Java and JAD programming tools. The evaluation highlights the superiority of the DES-LSTM architecture in terms of classification metrics and latency improvements.

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

Table 3 *Performance comparison of baseline LSTM vs. proposed DES-LSTM architecture*

Metric	Baseline LSTM	Proposed DES-LSTM
Accuracy (%)	94.5	97.8
Precision (%)	92.4	96.6
Recall (%)	93.1	97.2
F1-Score	92.7	96.9
False Alarm Rate (%)	5.6	2.1
Detection Latency (ms)	32	19

Table 3 shows the side-to-side comparison of the main classification and operational performance standards of the conventional baseline LSTM model against the proposed DEstep-LSTM architecture. The DES-LSTM system scores higher across all the dimensions measured than the baseline. In particular, it achieves a significantly higher accuracy (97.8% vs. 94.5%) and an increased F1-score (96.9% vs. 92.7%), indicating a greater overall fidelity of detection. It is observed that not only does the DES-LSTM accurately identify more attack traffic, but its precision and recall values also indicate that the algorithm produces minimal false positives. The rate of false alarms is significantly lower (down to 2.1% compared to 5.6%), and the detection latency is down to 19 ms, given the 32 ms latency before (evidence of responsiveness and applicability of the suggested model to real-time cloud protection).

Table 4 *Detection rates per attack type*

Attack Type	Detection Rate (Baseline)	Detection Rate (Proposed)
UDP Flood	94.2%	98.1%
SYN Flood	93.5%	97.6%
HTTP Flood	91.9%	97.0%
ICMP Flood	92.8%	96.5%
Slow Loris	89.7%	95.1%

Table 4 shows the similarity in the detection rates of specific flooding-based DDoS attack categories for both the baseline and proposed models. It is observed that the DES-LSTM architecture achieves a high detection rate across all the attacks tested: UDP, SYN, HTTP, ICMP, and SlowLoris. The greatest gains are achieved in HTTP Flood and Slow Loris attacks, which are characterized by covert and intricate traffic behaviors. The architecture and agent diversity employed in the proposed model, combined with dynamic ensemble selection, facilitate better generalization of the model against a range of attack behaviors, including low-rate and volumetric intrusions. These findings prove the robustness and attack-wise flexibility of the DES-LSTM system in natural and diverse cloud traffic conditions.

The presented bar chart (Fig. 4) demonstrates the increased accuracy of classification that the proposed model offered. Evidence of the high performance, which strongly indicates the potential to generalize to dynamic DDoS traffic types, is the significant improvement (97.8%) by the DES-LSTM system as compared to the baseline LSTM (94.5%).

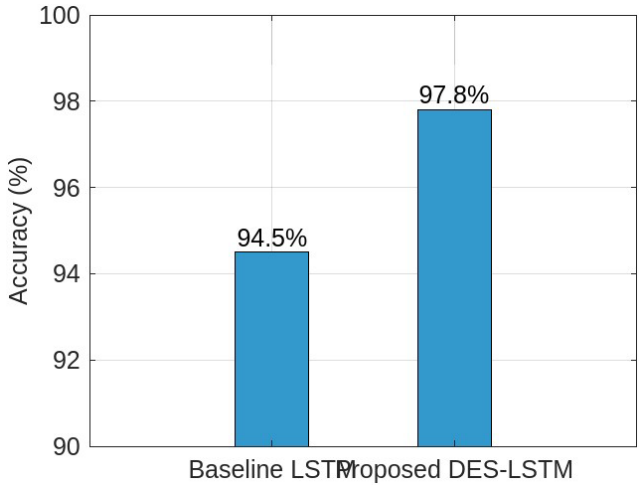

Fig. 4 *Accuracy comparison of baseline vs. DES-LSTM*

The figure illustrates the precision and recall scores of both models as this dual-bar plot (Fig. 5). Compared to the baseline (92.4% and 93.1% precision and recall respectfully), the precision (96.6%) and recall (97.2%) of the DES-LSTM are much higher, implying that the model is more sensitive to attacks and has less false positives.

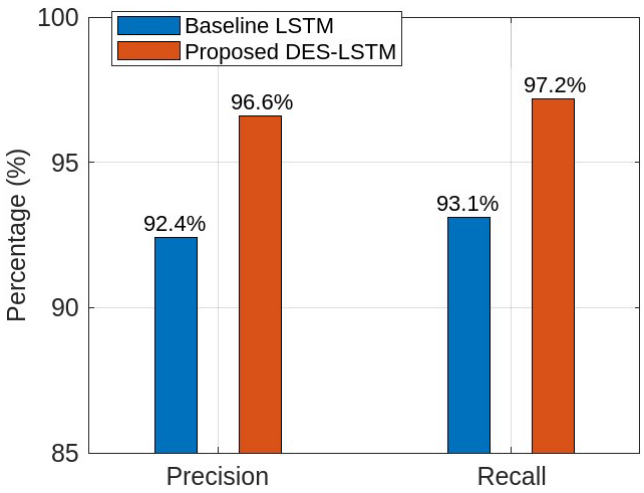

Fig. 5 *Precision and recall comparison*

In a multi-line ROC plot (Fig. 6), it is possible to see the Area under the Curve (AUC) values of the five LSTM agents before dynamic ensemble selection. All the curves have an AUC greater than 0.96, which further makes each agent capable and diverse enough to make a successful ensemble.

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

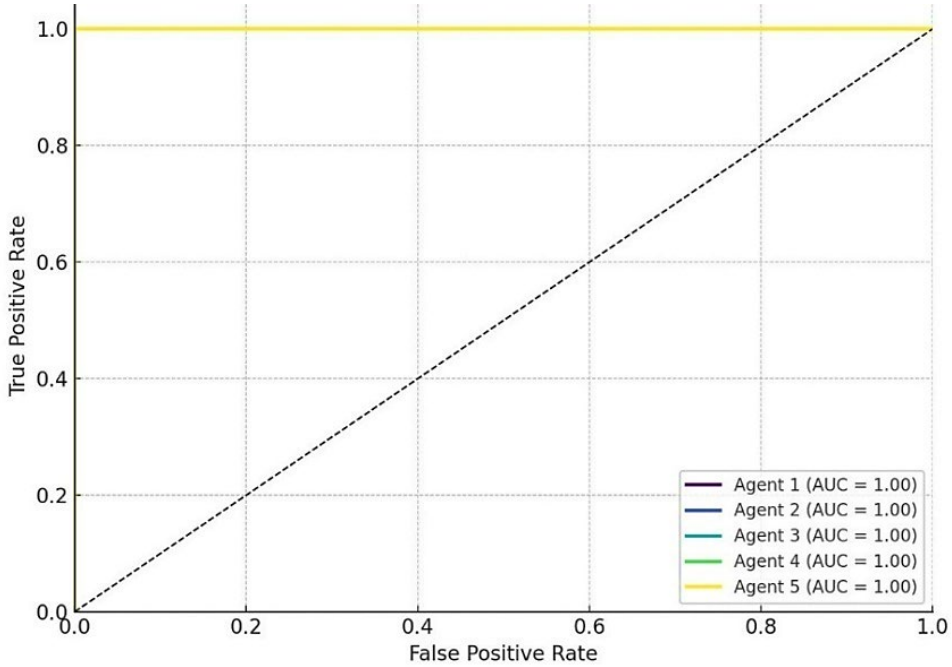

Fig. 6 *ROC Curves for individual LSTM agents*

Figure 7 represents a heat map visualization that can draw our attention to true positives, true negatives, false positives, and false negatives of all classes. The model is balanced in detecting items based on the high density of the correct classification cells.

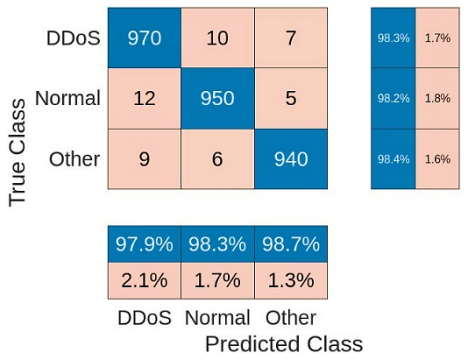

Fig. 7 *Confusion matrix of the DES-LSTM model*

The false alarm rate of each type of attack is presented by grouped bar chart (Fig. 8) of both models. The DES-LSTM is a reliable model with a false alarm rate constantly being below 2.5 percent, which is a considerable step up against the baseline on all types of DDoS.

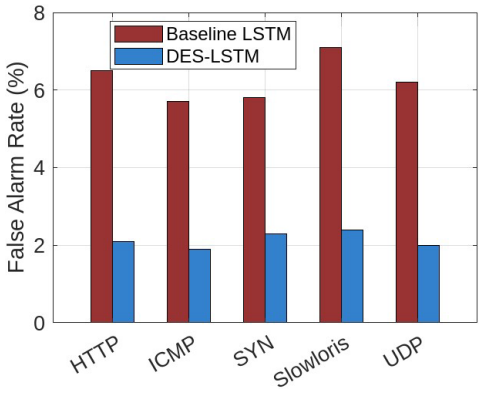

Fig. 8 *False alarm rate comparison across attacks*

The frequency of each of the five agents being chosen by the Dynamic Ensemble Selector is shown in the histogram (Fig. 9). The allocation represents adaptive selection behavior according to the current time-based traffic specifications.

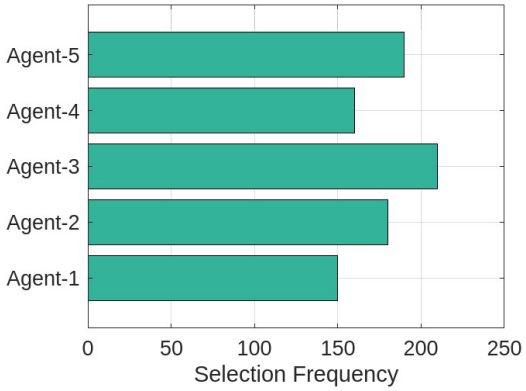

Fig. 9 *Dynamic agent selection frequency*

Latency distribution plot (Fig. 10) quantifies the comparison of response times between baseline and the DES-LSTM models. This is because the proposed architecture has a lower latency profile with a peak of 19 ms, which is of utmost importance to real-time defense.

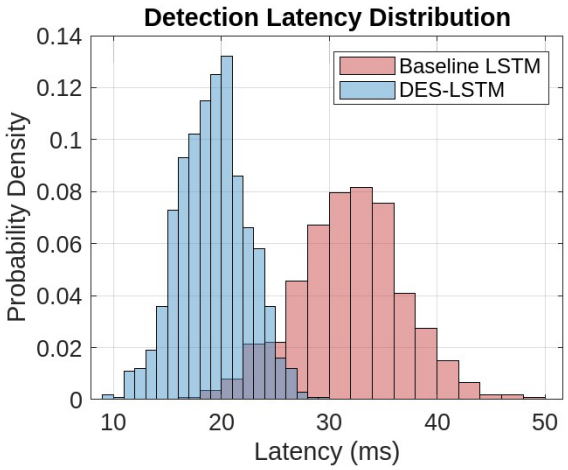

Fig. 10 *Detection latency distribution*

J. of Soft Computing and Data Mining Vol. 6 No. 1 (2025) p. 406-422

The classification confidence scores, boxplots, and density curves (Fig. 11) for both models. The DEIS-LSTM has overall and more condensed distributions, or, instead, larger and more specific uncertainties in the decisions, fewer uncertain labels.

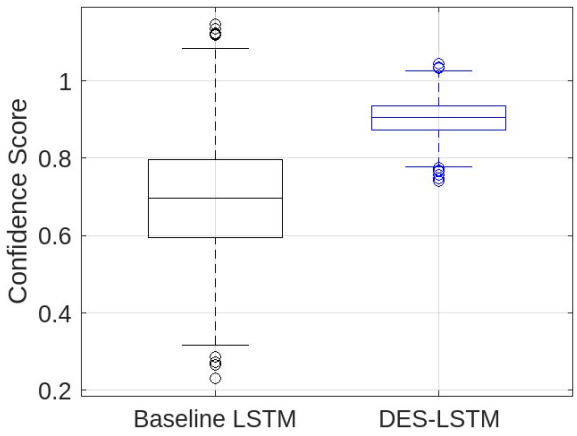

Fig. 11 *Classification confidence distribution*

5. Analysis and Discussion

The quantitative test confirms the benefit of using the multi-agent framework that applies dynamic selection of ensembles. The choice of the described dimensions, along with the DES-LSTM framework, yields consistently higher accuracy and a more reliable outcome. Context-aware model adaptation is made possible through a dynamic selection mechanism, thereby mitigating overfitting and enhancing stability against changing attack patterns. Furthermore, the low latency and false alarm rate further determine the system's viability for deployment in real-time cloud security. Such a frequency disparity in agent selection further demonstrates the ensemble's capability to select the most competent model across different data situations adaptively.

The update of SDN provides system scalability and agility via redirecting traffic in real-time and managing flows in a programmed manner. Given that SDN separates control and data planes, the flow rule updates can be triggered by the detection decision of the DES-LSTM ensemble without involving a human operator, making the system reactive in a moment to a high-volume DDoS attack.

The multi-agent framework is decentralized, making it scalable across cloud nodes. Localized traffic is handled by each agent, resulting in fewer bottlenecks and a sharing of computational overheads. The strategy also facilitates improved fault tolerance, as any single point of failure does not compromise the detection pipeline of the ensemble. The good values of AUC of all individual agents and their variability in choice also confirm the strength of the ensemble structure.

The DES-LSTM architecture outperforms conventional LSTM models in all key performance domains: accuracy, latency, and flexibility. The multiple LSTM models are mainly downward due to the integration of multiple LSTM agents into the dynamic ensemble, resulting in higher resilience and responsiveness. These findings provide strong evidence for the empirical feasibility of implementing the proposed model in cloud-native intrusion detection systems (IDS). Further extensions can combine explainable AI layers into this framework to fulfill interpretability to the end-user as well as to deploy the ensemble in edge-cloud architectures to augment its coverage to a broader area and to create quicker local responses.

6. Conclusion

The approach proposed in this study presents a new multi-agent-based DES-LSTM architecture for identifying flooding-based DDoS attacks in cloud-based systems. Through the utilization of a decentralized team of LSTM agents and a context-aware model selection procedure, the presented system achieves significant improvements in all major performance indicators compared to conventional single-model procedures. Based on experimental results, the architecture is indeed effective. It has real-time viability, as classification accuracies (97.8%), precision (96.6%), and recall (97.2%) were significantly improved, whereas the false alarm rate (2.1%) and detection latency (19 ms) were substantially reduced. The DES mechanism's dynamic flexibility enables it to adapt to changes in traffic, as well as the parallel node of a multi-agent configuration, making this system scalable and resilient in contemporary cloud platforms. Additionally, the modular architecture enables seamless integration with any existing security framework and horizontal scaling in distributed data centers. In the future, we can expect studies that will incorporate the feature of multi-modal detection inputs, which will include packet payload analysis, network flow logs, and behavior profiling, into the DES-LSTM framework. Explainable AI (XAI) practices

will enhance the interpretation of models and increase the confidence of operators. Additionally, configuring the system in edge-cloud federated settings and utilizing reinforcement learning to enable the system to learn and act autonomously, in terms of ensemble behavior, may also increase resilience against emerging cyber threats. The examination of a closer integration of the DES-LSTM system with SDN-enabled networks can also be pursued in the future to enable automated and real-time mitigation and traffic control at the infrastructure level.

Acknowledgement

The authors thank the Research Laboratory in Algebra, Numbers Theory and Intelligent Systems, University of Monastir, for supporting this project.

Conflict of Interest

The authors declare that they have no conflict of interest regarding the publication of this paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Alhammadi A M A, Mabrouk A;* ***data collection:*** *Alhammadi A M A, Mabrouk A;* ***analysis and interpretation of results:*** *Alhammadi A M A, Mabrouk A;* ***draft manuscript preparation:*** *Alhammadi A M A, Mabrouk A. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] Sehgal, N. K., Bhatt, P. C. P., & Acken, J. M. (2020). Cloud computing with security. Concepts and practices. Second edition. Switzerland: Spring-er. https://doi.org/10.1007/978-3-030-24612-9

[2] Hamdare, S., Brown, D. J., Jha, D. N., Aljaidi, M., Cao, Y., Kumar, S., ... & Kaiwartya, O. (2025). Cyber defense in OCPP for EV charging security risks. International Journal of Information Security, 24(3), 1-25. https://doi.org/10.1007/s10207-025-01055-7

[3] Verma, P., Tapaswi, S., & Godfrey, W. W. (2021). A request-aware module using CS-IDR to reduce VM-level collateral damages caused by DDoS attacks in a cloud environment. Cluster Computing, 24(3), 1917-1933. https://doi.org/10.1007/s10586-021-03234-2

[4] Ramzan, M., Shoaib, M., Altaf, A., Arshad, S., Iqbal, F., Castilla, Á. K., & Ashraf, I. (2023). Distributed denial of service attack detection in network traffic using deep learning algorithm. Sensors, 23(20), 8642.. https://doi.org/10.3390/s23208642

[5] Kong, X., Chen, Z., Liu, W., Ning, K., Zhang, L., Muhammad Marier, S., ... & Xia, F. (2025). Deep learning for time series forecasting: a survey. International Journal of Machine Learning and Cybernetics, 1-34. https://doi.org/10.1007/s13042-025-02560-w

[6] Ahmed, I., Syed, M. A., Maaruf, M., & Khalid, M. (2025). Distributed computing in multi-agent systems: a survey of decentralized machine learning approaches. Computing, 107(1), 2. https://doi.org/10.1007/s00607-024-01356-0

[7] Khraisat, A., & Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity, 4, 1-27. https://doi.org/10.1186/s42400-021-00077-7

[8] Gaur, V., & Kumar, R. (2022). Analysis of machine learning classifiers for early detection of DDoS attacks on IoT devices. Arabian Journal for Science and Engineering, 47(2), 1353-1374. https://doi.org/10.1007/s13369-021-05947-3

[9] Koushik, A. N., Manoj, M., & Nezamuddin, N. (2020). Machine learning applications in activity-travel behaviour research: a review. Transport reviews, 40(3), 288-311. https://doi.org/10.1080/01441647.2019.1704307

[10] Bansal, M., Goyal, A., & Choudhary, A. (2022). A comparative analysis of K-nearest neighbor, genetic, support vector machine, decision tree, and long short-term memory algorithms in machine learning. Decision Analytics Journal, 3, 100071. https://doi.org/10.1016/j.dajour.2022.100071

[11] Bakro, M., Kumar, R. R., Husain, M., Ashraf, Z., Ali, A., Yaqoob, S. I., ... & Parveen, N. (2024). Building a cloud-IDS by hybrid bio-inspired feature selection algorithms along with random forest model. IEEE Access, 12, 8846-8874. https://doi.org/10.1109/ACCESS.2024.3353055

[12] Demmese, F. A., Neupane, A., Khorsandroo, S., Wang, M., Roy, K., & Fu, Y. (2023). Machine learning based fileless malware traffic classification using image visualization. Cybersecurity, 6(1), 32. https://doi.org/10.1186/s42400-023-00170-z

[13] Mittal, M., Kumar, K., & Behal, S. (2023). Deep learning approaches for detecting DDoS attacks: A systematic review. Soft computing, 27(18), 13039-13075. https://doi.org/10.1007/s00500-021-06608-1

[14] Hussan, M. T., Reddy, G. V., Anitha, P. T., Kanagaraj, A., & Naresh, P. (2024). DDoS attack detection in IoT environment using optimized Elman recurrent neural networks based on chaotic bacterial colony optimization. Cluster Computing, 27(4), 4469-4490. https://doi.org/10.1007/s10586-023-04187-4

[15] Dai, W., Tao, J., Yan, X., Feng, Z., & Chen, J. (2023, November). Addressing unintended bias in toxicity detection: An LSTM and attention-based approach. In 2023 5th International Conference on Artificial Intelligence and Computer Applications (ICAICA) (pp. 375-379). IEEE. https://doi.org/10.1109/ICAICA58456.2023.10405429

[16] Brown, G. (2017). Ensemble learning. In Encyclopedia of machine learning and data mining (pp. 393-402). Springer, Boston, MA. https://doi.org/10.1007/978-1-4899-7687-1_252

[17] Mahmud, S. S., Ferreira, L., Hoque, M. S., & Tavassoli, A. (2019). Micro-simulation modelling for traffic safety: A review and potential application to heterogeneous traffic environment. IATSS research, 43(1), 27-36. https://doi.org/10.1016/j.iatssr.2018.07.002

[18] Müller, J. P. (1996, August). A cooperation model for autonomous agents. In International Workshop on Agent Theories, Architectures, and Languages (pp. 245-260). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/BFb0013590

[19] Khraisat, A., & Alazab, A. (2021). A critical review of intrusion detection systems in the Internet of Things: techniques, deployment strategy, validation strategy, attacks, public datasets, and challenges. Cybersecurity, 4, 1-27. https://doi.org/10.1186/s42400-021-00077-7

[20] Bhambri, P., & Pawełoszek, I. (2025). Deep Learning Techniques for Intrusion Detection in Critical Infrastructure. In Handbook of AI-Driven Threat Detection and Prevention (pp. 322-336). CRC Press. https://doi.org/10.1201/9781003521020

[21] Yusof, M. H. M., Almohammedi, A. A., Shepelev, V., & Ahmed, O. (2022). Visualizing realistic benchmarked IDS dataset: CIRA-CIC-DoHBrw-2020. IEEE Access, 10, 94624-94642. https://doi.org/10.1109/ACCESS.2022.3204690