International Journal of
Computational Learning & Intelligence
Vol. 04 | Issue 04 | 2025

RESEARCH ARTICLE OPEN ACCESS

Deep Learning Approach for IoT Traffic Multi-Classification in a Smart-City Scenario

G Ramasubba Reddy[1] . **Sunil J**[1] . **S Nareshkumar Reddy**[1] . **L Jayasree**[2] . **T V N Radha Parameswari**[1]

[1]Department of CSM, Sai Rajeswari Institute of Technology, Proddatur, India.
[2]Department of CSE, Sri Padmavathi Mahila Vishwavidyalaya, Tirupati, India.

DOI: **10.5281/zenodo.17376640**
Received: 14 September 2025 / Revised: 14 October 2025 / Accepted: 17 October 2025
©Milestone Research Publications, Part of CLOCKSS archiving

Abstract – The recent explosive growth of Internet of Things (IoT) devices in smart cities has grown the attack surface of modern networks exponentially, calling for efficient and scalable intrusion detection means. Classical rule-based and classical machine learning (ML) approaches are typically not able to handle the heterogeneity and dynamic nature of IoT traffic. We recommend Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks combined in a hybrid deep learning (DL) model to effectively capture spatial and temporal dependencies in network flow data in this work. We preprocess ACI-IoT-2023 dataset with over 1.23 million records of benign and malware traffic through feature encoding, Min-Max normalization, and feature selection in order to present the inputs as balanced and optimized. Experimental results confirm that the suggested CNN-LSTM model performs superior with improved classification accuracy of 99.99% on average and near-perfect precision, recall, and F1-measures for every attack type. Comparison to traditional baselines including Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), CNN, and LSTM indicates the robustness and durability of the proposed method. These results indicate that hybrid CNN-LSTM is a strong contender for real-time IoT intrusion detection in the context of smart cities.

Index Terms – IoT Security, Intrusion Detection System, DL, ML, CNN-LSTM, Smart City Networks, Network Traffic Classification

I. INTRODUCTION

In today's smart cities, the IoT is growing rapidly in energy, transportation, healthcare, and surveillance. Secure and reliable communication has become a primitive challenge due to the massive volumes of heterogeneous traffic sent by billions of networked devices [1]. Innovative city architecture mainly relies on real-time IoT traffic classification to identify malicious activity and prevent disruptions in essential services [2]. Traditional threshold-based or static rule-based approaches are insufficient for IoT systems because they are dynamic and resource-constrained, unlike conventional networks [3]. However, it is still challenging to classify IoT traffic into multiple classes for several reasons. For example, numerous protocols, data structures, and communication layers are involved in IoT traffic, resulting in intricate and interconnected patterns [4]. Then, conventional cyberattacks that imitate regular traffic, like DDoS, botnets, and spoofing, necessitate high-fidelity feature extraction that goes beyond basic heuristics [5]. Again, hand-crafted or computationally costly solutions cannot be adopted due to the scalability requirements of smart-city networks [6]. The development of computerized, intelligent models that can adapt to changing network infrastructures has been spurred by these challenges.

In recent years, many researchers have used ML, and ensemble approaches to detect IoT intrusions. In earlier works, SVMs, k-Nearest Neighbors (k-NN), DTs, RFs, and boosting architectures have exhibited promise [7], [8]. Optimization-inspired strategies, such as Whale Optimization Algorithms, Particle Swarm Optimization, and Genetic Algorithms, have been applied to ML classifiers to assist with feature selection and convergence [6], [7]. While these approaches were more precise in controlled environments, they are still afflicted with substantial limitations: reliance on hand-engineered features, inability to be robust to cross-domain traffic, slow inference speed, and poor accuracy on multi-class attack examples [3], [2]. With hierarchical temporal and spatial features self-learned from raw or preprocessed data streams, DL has started to transform IoT traffic analysis. The temporal and spatial correlations in network traffic dependencies can be learned by Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and their hybrid CNN-LSTM models [4]. Different types of advanced techniques, such as hybrid optimization-DL models and graph neural networks (GNNs), have shown promise in simulating dynamic traffic flows and device-to-device interactions [7], [5],[6]. There is still a lot to be done, however: poor generalization in diverse IoT deployments, too many false positives for multi-class detection, and limited benchmarking in real-world smart-city settings. Motivated by the shortcomings of past work, this paper proposes a deep learning multi-classification framework for smart city IoT traffic. To capture local spatial patterns and sequential temporal behaviors of IoT traffic flows, our solution employs CNN-LSTM layers. Our framework also involves regularization and feature refinement techniques, which enhance generalization and alleviate overfitting, thereby increasing robustness. In contrast to conventional methods, the system developed here can automatically identify discriminative patterns from composite traffic without the need for feature engineering, guaranteeing adaptability and flexibility in changing attack situations.

The main contributions of this work are the following:

- We propose a CNN-LSTM DL architecture designed explicitly for multi-classification of IoT traffic within innovative city environments.
- We compare our method against standard ML and hybrid methods on widely used IoT intrusion datasets.
- We combine feature selection and regularization techniques to enhance robustness within multi-class contexts.

- We evaluate computational efficiency and model interpretability to ensure the practical feasibility of our solutions within smart-city infrastructures.

II. LITERATURE SURVEY

Large amounts of diverse traffic have been caused by the quick expansion of IoT devices in smart cities. Appropriate traffic classification is essential for network management, anomaly detection, and cyberattack prevention. However, IoT environments pose several challenges, including dynamic traffic patterns, encrypted communications, class imbalance, and constrained device resources. As a result, researchers have investigated various ML and DL approaches, followed by more recent transformer-based models and hybrid frameworks, to achieve robust IoT traffic classification and intrusion detection. Conti et al. [9] focused on enhancing IoT security in smart cities by developing a honeypot-driven ML framework. Their methodology reaped attacker activity through decoys and employed ML classifiers to augment training data. Results were improved attack detection, but the process is limited by its dependence on honeypot deployment and inability to scale to real-time multi-class traffic classification.

Xie et al. [10] attempted to minimize the computational expenses in IoT traffic classification through the application of knowledge distillation. Following compression and reduction of large teacher models, student models were trained more on benchmark IoT datasets. Although less generalized, accuracy remained constant in highly dynamic traffic. Liao and Guan [11] proposed a multi-scale convolutional feature fusion network with CBAM attention (MCF-CBAM) for classifying IoT traffic. Their approach improved accuracy by eliminating noisy features from traditional IoT datasets. However, the model's capacity to generalize to novel traffic patterns is constrained by convolutional architectures. Afifi et al. [12] addressed the issue of generalization by proposing MIND-IoT, a transformer-based tokenization model that utilizes CNNs. Their proprietary IoT-Tokenize pipeline preserved traffic semantics and was pre-trained on enormous datasets (UNSW IoT Traces and MonIoTr) with subsequent fine-tuning on IoT Sentinel and YourThings. The model's exceptional cross-dataset adaptability was demonstrated by its accuracy of over 98%. However, the approach is computationally intensive, making it impractical to implement in limited IoT edge devices.

Ismail et al. [13] demonstrated lightweight ML algorithms for intrusion detection using the TON_IoT, WUSTL-IIoT-2021, and Edge-IIoT datasets. DTs, RFs, and LightGBM were compared, and ensemble approaches were found to have the best trade-off between accuracy and efficiency. However, their models were still dataset-imbalance-sensitive and did not generalize well to unseen traffic classes. Zahid and Bharati [14] proposed a hybrid CNN–BiLSTM model for detecting IoT attacks. Trained on benchmark datasets (KDDCup99, NSL-KDD, and CIC-IDS2017), the method achieved an accuracy of up to 99.9% by combining spatial and temporal feature learning. Although achieving strong outcomes, there are weak aspects, including high training complexity and reduced explainability. Miao and Liao combined CNN and Particle Swarm Optimization (PSO) [15] to intelligently predict city traffic. CNN hyperparameters were dynamically optimized using data from IoT sensors. They improved traffic flow efficiency by 25% and decreased congestion by 20%. Scalability is the main drawback for big, diverse networks. Andrysiak et al. [16] presented GC-YOLOv9, which incorporated Ghost Convolution into YOLOv9 for smart city traffic monitoring. With testing conducted on the BDD100K and Cityscapes datasets, the model increased detection accuracy in real-world traffic environments with high complexity. Limitations, however, are its lower inference rates for real-time processing with high data rates.

For all such categories, traditional ML and CNN models offer lightweight and efficient solutions; however, they are not generalized across diverse traffic. Transformer models are very flexible but plagued by computational and deployment problems. Hybrid and ensemble models are accurate, but they are either too complex or non-interpretable. Domain-specific extensions in an application domain, such as GC-YOLOv9, offer better detection precision but are plagued by real-time scalability challenges. These works altogether demonstrate that robust multi-class IoT traffic classification in smart cities is an open issue due to the trade-off between accuracy, efficiency, and adaptability. We offer a DL system for IoT traffic multi-classification that fills the gap by combining real-time adaptability, flexible model design, and sophisticated feature extraction, providing a comprehensive solution for creative city applications.

III. METHODS & MATERIALS

The methodology for analysing the paper follows a series of steps that range from data acquisition and preprocessing to feature selection, baseline model evaluation, and the development of a novel CNN+LSTM hybrid deep network architecture. Each step is chosen in order to preserve reproducibility, reliability, and peak performance in IoT traffic multi-classification in a smart-city environment. Figure 1 represents the overall research methodology.

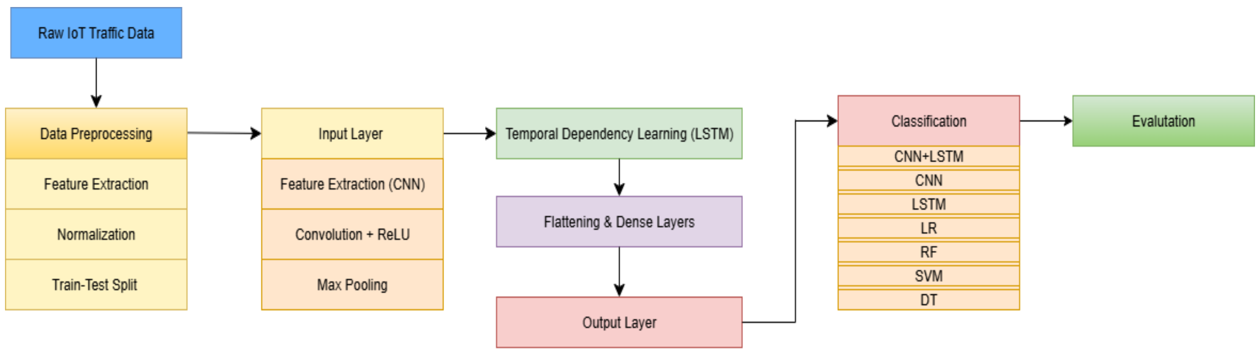

Fig. 1: Graphical representation of the overall research methodology

A. Dataset Description

The dataset utilized here in the study is the ACI-IoT-2023 dataset[17], consisting of 1,231,411 records of network flow traffic collected by various IoT devices located in a smart-city environment. The devices are smart cameras, sensors, thermostats, and voice assistants, and both wired and wireless traffic is emitted from them. Each network flow is labelled as benign or belonging to one of several categories of attacks, like Port Scan, ICMP Flood, DNS Flood, and other types of intrusion. The data set captures detailed flow-level statistics, such as packet counts, packet sizes, inter-arrival times, and TCP/UDP flags, and provides a detailed impression of network activity. These factors are crucial in distinguishing normal vs. malicious network traffic patterns in IoT networks. For a better overview, Table 1 shows some of the most important features selected from the data set:

Table 1: Summarizing the features of the Dataset

Column Name	Description
Src IP	Source device IP address
Dst IP	Destination device IP address
Src Port	Source port number
Dst Port	Destination port number
Protocol	Network protocol used (TCP, UDP, ICMP, etc.)
Flow Duration	Duration of the network flow in seconds
Total Fwd Packets	Total number of packets sent in the forward direction
Total Bwd Packets	Total number of packets sent in the backward direction
Fwd Packet Length Mean	Average size of packets in the forward direction
Bwd Packet Length Std	Standard deviation of packet sizes in the backward direction

B. Data Preprocessing

Accurate and efficient preprocessing is crucial for multi-class IoT traffic classification. The ACI-IoT-2023 dataset consists of numerical and categorical features along with different scales and potential missing values. The preprocessing pipeline was utilized for consistency, noise elimination, and preparing high-quality input for classical and DL models.

1. Handling Missing and Irrelevant Values: Some features had missing, infinite, or undefined values due to measurement errors or division by zero, particularly Flow Bytes/s and Flow Packets/s. These columns were removed to avoid incorrect calculations. Features indirectly beneficial for traffic classification, such as Timestamp, were also removed. This avoids the model focusing on traffic behaviour rather than uninformative metadata for attack detection.

2. Categorical Encoding: Categorical features such as Protocol, Label, and Connection Type were represented as numbers by using Label Encoding, mapping each unique category to an integer of distinct value:

$$y_i' = LabelEncode(y_i)$$

Where y_i is the original categorical value and y_i' is the encoded numerical value. This mapping helps ML and DL models handle categorical inputs efficiently.

3. Feature Normalization: IoT network traffic patterns are extremely diverse in scale, e.g., packet lengths vary from a few bytes to thousands, whereas inter-arrival times vary from milliseconds to seconds. To achieve numerical stability and better convergence of the models, all continuous features were normalized using Min-Max scaling:

$$x_i' = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Where x_i is the original feature value, x_{min} and x_{max} are the minimum and maximum values of the feature across the dataset, and x_i' is the normalized value within the range [0, 1] bounded. This normalization process prevents the learning process from being dominated by features with large magnitudes.

4. Train-Test Split: For estimating model performance, both class-balanced and stable, the data set was split stratified by class labels into training and test subsets of an 80/20 ratio. Stratified splitting preserves the original distribution of attack types in both subsets, which is essential in consideration of the highly unbalanced classes.

5. Feature Selection: Since the dataset is high-dimensional, noise or redundant features can cause higher computational cost and lower model generalizability. Two methods were applied, complementary to each other, for feature selection:
Recursive Feature Elimination (RFE): Repeatedly trains a classifier and removes the least significant features according to model weights until reaching the desired number of features.
Correlation Analysis: Pearson correlation coefficients are calculated between pairs of features:
$$r_{ij} = \frac{conv(\ x_i, x_j)}{\sigma_{x_i}\sigma_{x_j}}$$
Where $conv(\ x_i, x_j)$ is the covariance between features $x_i \ and \ x_j$ and σ_{x_i} is the standard deviation of x_i. Features with $|r_{ij}| > 0.9$ were excluded to avoid multicollinearity.
The chosen feature vector for every flow that was achieved is represented as:
$$X = [x_1, x_{2, \ldots\ldots\ldots,} x_n]\ x_i \epsilon \mathbb{R}$$
where n is the number of chosen features.

6. Handling Imbalanced Classes: IoT traffic data sets exhibit extremely skewed class imbalance, i.e., rare attacks like ARP Spoofing vs. common Port Scan occurrence. To counter this: Stratified Sampling: Provides proportionate representation of all classes in the training and test sets.

B. Methodology

1. Baseline Models: To provide a comparison baseline, several ML models were employed as baseline classifiers for the multi-classification of IoT traffic. The models were selected based on their proven suitability for intrusion detection and multi-class classification tasks.

   - **Logistic Regression (LR):** LR was employed as a linear baseline model, an interpretable and lightweight classifier. LR is utilized to predict the likelihood of a traffic flow being in a specific attack or benign class through a SoftMax function for multi-class classification.

   $$P(y = c|x) = \frac{\exp\ (\beta_c^T x)}{\sum_{k=1}^{K} (\beta_k^T x)}$$

   where x is the feature vector, β_c is the parameter vector of class c, and K is the number of classes.

   - **Convolutional Neural Network (CNN):** CNNs are good at learning local spatial patterns in sequential data automatically through convolutional filters. In the baseline CNN of this study, one-dimensional convolutional layers were applied to learn discriminative fault-related features. The structure of the model consisted of convolutional layers followed by max-pooling to progressively reduce dimensionality and enhance feature abstraction. Then, densely connected layers with ReLU activation were applied before the final softmax output layer for classification. The CNN baseline was principally employed as a feature extractor to benchmark the performance of spatial pattern recognition alone [18].

- **Long Short-Term Memory (LSTM):** LSTMs are a variant of a recurrent neural network (RNN) that can learn long-term time series data dependencies by circumventing the vanishing gradient problem using memory cells and gating mechanisms. The LSTM baseline for this study was constructed with stacked LSTM layers to understand the temporal dependencies naturally present. Hidden and cell states across time were retained by each LSTM cell such that the model had the ability to capture useful temporal information. The output step employed a dense layer with softmax activation in order to generate class probabilities[19].
- **Support Vector Machine (SVM):** SVM was trained with a radial basis function (RBF) kernel to deal with nonlinear class boundaries in high-dimensional space [20]. The decision function is expressed as:

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i y_i \, k(x_i, x) + b\right)$$

Where $k(x_i, x)$ is the kernel function, α_i is the learned coefficients, and y_i is the class labels.

- **Random Forest (RF):** To prevent overfitting and model feature interactions, RF, an ensemble of DTs, was used. The predictions were made by averaging the votes from individual trees, and the final class was determined using majority voting[21].
- **Decision Tree (DT)**: DTs were employed to evaluate the classification ability of rule-based hierarchical models [22]. Each traffic flow is classified by splitting features at decision nodes based on Gini impurity:

$$G = 1 - \sum_{i=1}^{C} p_i^2$$

where p_i is the probability that a sample belongs to class i.

2. Proposed CNN+LSTM Hybrid Architecture: To overcome the limitations of traditional models in learning temporal and spatial dependences of IoT traffic, a hybrid DL architecture with CNN and LSTM networks was proposed [23].

- Input Layer: Input is preprocessed feature vectors of individual IoT traffic streams. Features are scaled to [0, 1] to offer equal scaling.
- CNN Block: The CNN block derives local spatial dependencies between corresponding traffic features. Various 1D convolutional filters along feature directions generate low- and high-level features. Convolutional operation in both instances, followed by ReLU activation and max pooling, is applied to reduce dimensionality.

$$h_j^l = f(\sum_{i=1}^{n} w_{ij}^l . x_i + \ b_j^l)$$

where h_j^l is the activation of the j-th filter in layer l, w_{ij}^l is the weight matrix, and f is the ReLU activation.

- LSTM Block (Temporal Dependency Modeling): The output of the CNN block is passed to stacked LSTM layers for capturing sequential dependencies from traffic flows. The LSTM updates are regulated by the forget, input, and output gates:

$$f_t = \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right)$$

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

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

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh\ (W_c . [h_{t-1}, x_t] + b_c)$$

$$h_t = o_t \odot \tanh\ (c_t)$$

  where f_t, i_t, o_t are the forget, input, and output gates, respectively; c_t is the cell state; and h_t is the hidden state.

- Fully Connected Layer: After the hierarchical spatial–temporal features are learnt using the CNN and LSTM layers, the resulting feature map is flattened into a one-dimensional vector for dense connectivity purposes. The flat structure preserves both the local discriminative patterns learnt by the CNN and the long-range dependencies learnt by the LSTM. The vector is then sent to a single or multiple dense, fully connected layers, in which every neuron learns a weighted sum of the features in order to supply intricate non-linear decision boundaries. Dropout regularization is employed to prevent overfitting by randomly turning off a specified percentage of the neurons during training, such that the network becomes independent of specific feature paths. Batch normalization is also introduced to stabilize and accelerate convergence by normalizing the layer activations, improving the generalization performance across different classes of IoT traffic.
- Output Layer: The final dense layer is linked to the output layer, having the same number of neurons as traffic classes (kinds of attacks and benign). In this place, a SoftMax activation is applied, which maps raw outputs (logits) to a normalized probability for each class. This ensures that the probabilities sum up to 1, and the maximum probability is the outputted traffic class. Mathematically, for any class j, probability equals:

$$P(y = j|x) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

where z_j represents the class j, and C denotes the number of classes. The probabilistic interpretation enables multi-class classification in the smart-city IoT scenario so that the model can accurately classify between normal traffic and different types of malicious attacks. The softmax output is then used for categorical cross-entropy loss computation, which guides the network parameter updates during training.

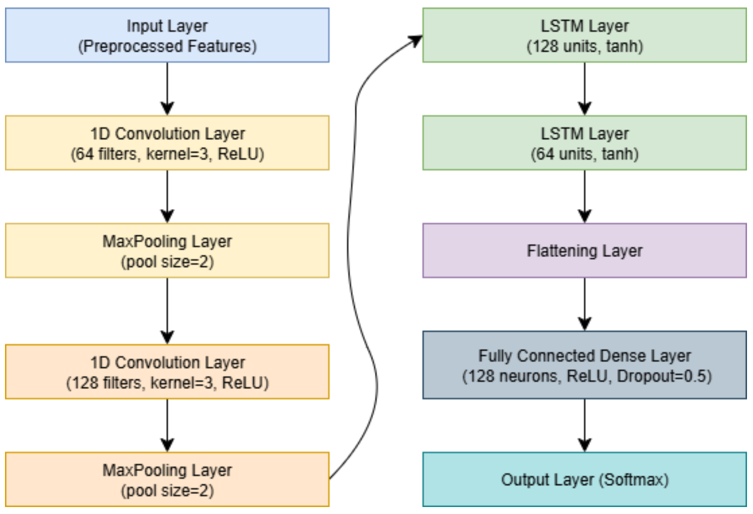
Input Layer (Preprocessed Features)
1D Convolution Layer (64 filters, kernel=3, ReLU)
MaxPooling Layer (pool size=2)
1D Convolution Layer (128 filters, kernel=3, ReLU)
MaxPooling Layer (pool size=2)
LSTM Layer (128 units, tanh)
LSTM Layer (64 units, tanh)
Flattening Layer
Fully Connected Dense Layer (128 neurons, ReLU, Dropout=0.5)
Output Layer (Softmax)

Fig. 2: Overview of the CNN+LSTM architecture

- Loss Function and Optimization: Class-Weighted Loss: While training DL, a weighted categorical cross-entropy loss was employed:

$$\mathcal{L}_{weighted} = -\sum_{i=1}^{C} w_i . y_i \log(\hat{y}_i)$$

Where:

$$w_i = \frac{N}{C . N_i}$$

Here, C is the number of classes, N is the total number of samples, N_i is the number of samples in class i, y_i is the ground truth label (one-hot encoded), $\hat{y}_i$ is the predicted probability for class i.

- Model weights are updated using the **Adam optimizer,** a variant of stochastic gradient descent that combines adaptive learning rate with momentum:

$$\theta_{t+1} = \theta_t - \eta . \frac{\widehat{m_t}}{\sqrt{\hat{v}_t} + \epsilon}$$

Where, $\widehat{m_t}$ and $\hat{v}_t$ are bias-corrected first and second moment estimates, and η is the learning rate (set to 0.001).

Table 2: Proposed Model Configuration and Training Strategy

Component	Specification
Input Layer	Preprocessed IoT traffic features (normalized)
Convolutional Layers	Two 1D CNN layers with 64 and 128 filters, kernel size = 3, stride = 1, ReLU activation
Pooling Layers	MaxPooling1D after each convolution (pool size = 2)
LSTM Layers	Two stacked LSTM layers with 128 and 64 hidden units, tanh activation
Flattening Layer	Converts multidimensional feature maps to 1D representation
Fully Connected Layer	Dense layer with 128 neurons, ReLU activation, Dropout rate = 0.5
Output Layer	Dense layer with softmax activation for multi-class classification

Optimizer	Adam optimizer with learning rate = 0.001
Loss Function	categorical cross-entropy
Batch Size	32
Epochs	50(early stop)
Evaluation Metrics	Accuracy, Precision, Recall, F1-Score
Component	Specification
Input Layer	Preprocessed IoT traffic features (normalized)
Convolutional Layers	Two 1D CNN layers with 64 and 128 filters, kernel size = 3, stride = 1, ReLU activation

IV. RESULTS & DISCUSSION

A. Experimental Setup

Experimental evaluation was conducted using the ACI-IoT-2023 dataset, which holds almost 1,231,411 network traffic flows representing benign and attack classes. Every flow is characterized by 66 features that are extracted based on statistical, protocol-based, and traffic behavior properties. Redundant or incorrect features, such as Flow Bytes/s, Flow Packets/s, and Timestamp, were removed during preprocessing since they contained missing or infinite values. To remove feature redundancy, Recursive Feature Elimination (RFE) and correlation-based filtering were applied jointly, and this resulted in a better feature subset. Also, the categorical labels were encoded using one-hot encoding to enable multi-class classification. The dataset was partitioned randomly into training (80%) and test (20%) sets to ensure class distribution preservation. Class weighting in the loss function helped to balance class imbalance. Input features were scaled [0,1] using Min-Max scaling before training. The proposed hybrid architecture consisted of two 1D convolutional layers (64, 128 filters, kernel size = 3, ReLU activation, max pooling = 2) and two stacked LSTM layers (128, 64 units). Sequencing output was passed into a dense layer of 128 neurons with ReLU activation, followed by a dropout layer (rate = 0.5) to prevent overfitting. The output layer employed SoftMax activation for multi-class prediction. Model training was conducted using the Adam optimizer with a learning rate of 0.001. A weighted categorical cross-entropy loss function was used to deal with minority-majority classes. The batch size was fixed at 32, and early stopping with respect to validation loss with patience was used to prevent overfitting.

B. Evaluation Metrics

Accuracy, Precision, Recall, and F1-score were several evaluation metrics that were utilized to assess the performance of the models. True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) are the terms used to explain the categorization results.

Accuracy:
$$\text{Accuracy:} \frac{(\mathrm{TP} + \mathrm{TN})}{(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})}$$

Precision:
$$\text{Precision:} \frac{(\mathrm{TP}\ )}{(\mathrm{TP} +\ \mathrm{FP}\ )}$$

Recall:
$$\text{Recall:} \frac{(\mathrm{TP}\ )}{(\mathrm{TP} +\ \mathrm{FN}\ )}$$

F1-Score:
$$\mathrm{F1} - \text{score:}\ \frac{2 \times (\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall}}$$

C. Performance of the Models

Among all evaluated models, the proposed CNN+LSTM hybrid model demonstrated the most robust and consistent performance across all evaluation metrics, as shown in Table 3 and Figure 3. The performance of the proposed CNN+LSTM hybrid model was compared with traditional ML classifiers, including LR, SVM, DT, and RF, and DL models, CNN, and LSTM. Of the baselines, LR yielded a 91.2% overall accuracy, which indicates poor ability to capture the complex nonlinear relationships in IoT traffic but fair performance on linearly separable patterns. SVM outperformed LR at 93.1% accuracy using kernel functions to transform the data into higher dimensions to address nonlinearity. DT, a partition model of features using trees, achieved 92.5% accuracy but was overfitting, especially on minority classes. RF, an ensemble of numerous DTs, worked best with the baseline at 94.3% accuracy and added generalization and stability. In regard to DL baselines, the isolated CNN achieved an accuracy of 98.65%, which was an enormous leap compared to all the traditional baselines. This level of high accuracy is justified in CNN's applicability in extracting spatial feature representation of traffic flows within networks. Similarly, the model of LSTM achieved an accuracy of 98.23%, demonstrating applicability in extracting temporal dependencies among traffic sequences. But each of the models separately was still worse than the proposed hybrid, since CNN is itself not capable of comprehending sequential behavior fully, and LSTM is itself not capable of discovering optimal local spatial correlations.

Conversely, the proposed hybrid CNN+LSTM model achieved an almost perfect accuracy of 99.99%, well above all of the baseline models. The model also achieved very high precision (99.98%), recall (99.99%), and F1-score (99.98%), reflecting its ability to correctly classify both malicious and normal traffic into the different categories. It is because of the design of the model: convolutional layers effectively extract spatial patterns from flow-level features, and LSTM layers find temporal relationships between successive streams of traffic. Furthermore, the use of class-weighted loss functions and stratified training reduced the impact of class imbalance so that minority attack classes were well-classified. These results demonstrate CNN+LSTM hybrid architecture as a highly scalable and effective solution for real-time multi-class IoT traffic classification for smart-city networks, referencing the advantages of deep spatial and temporal feature fusion over traditional ML solutions.

Table 3: Performance of the Proposed and Baseline Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Proposed	99.99	99.98	99.98	99.98
CNN	98.65	98.60	98.70	98.65
LSTM	98.23	98.20	98.25	98.23
LR	91.2	91.2	91.2	91.2
SVM	93.1	93.1	93.1	93.1
RF	94.3	94.3	94.3	94.3
DT	92.5	92.5	92.5	92.5

D. Class-wise Performance Analysis

The class-wise plot in Figure 5 indicates the per-class accuracy, precision, recall, and F1-score of the twelve classes separately. The class-wise performance plot provides an overall view of per-class Precision, Recall, F1-score, and Accuracy, showing not only the strengths but also the subtle weaknesses of the model in processing individual classes. The model achieves an overall accuracy of 99.99%, indicating its great capability in differentiating among different classes of network traffic under various types of attacks. For the majority of classes, including Benign, DNS Flood, Dictionary Attack, ICMP

Flood, OS Scan, Ping Sweep, Port Scan, SYN Flood, Slowloris, and Vulnerability Scan, the model performs flawlessly with Precision, Recall, and F1-scores of 1.0. This shows that the model is very good at accurately predicting positive instances while also retrieving nearly all true instances for these well-represented classes, demonstrating excellent generalization and minimal misclassification. A slight deviation is observed in the UDP Flood class. Although Precision remains perfect at 1.0, the Recall is somewhat reduced (0.89), leading to an F1-score of 0.94. This is because some samples of UDP Flood are mixed up with ARP Spoofing, indicating mild confusion between these two classes. The ARP Spoofing class, however, is the least identified of all the classes. Its weakest performance is because of a relatively small number of samples and collision with UDP Flood predictions, demonstrating the fine-grained impact of class representation on model performance.

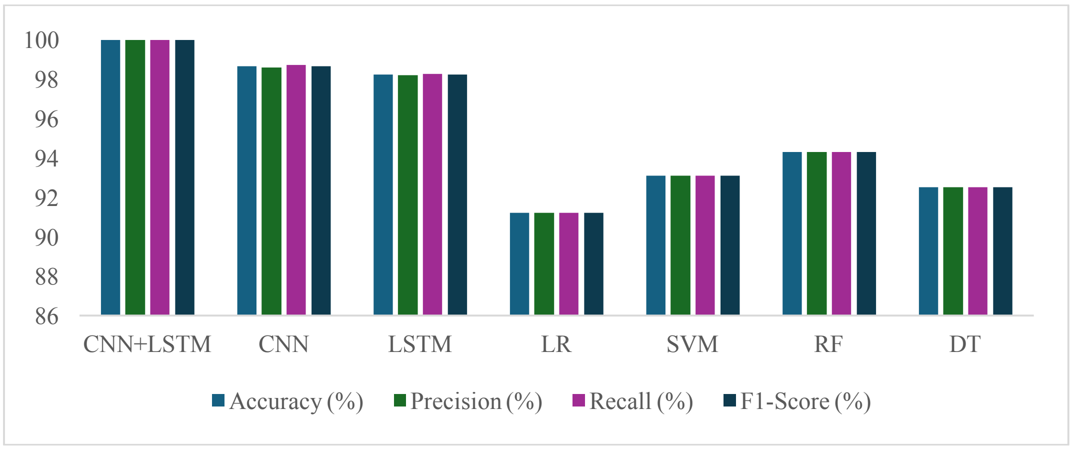

Fig. 3: Performance of the models

Though these minute differences are present, the model is extremely close to exactly identifying nearly all classes with an overall Accuracy of 99.99% and extremely high weighted-average statistics. The minute level of confusion between UDP Flood and ARP Spoofing means that class-weighted loss functions or class-specific data augmentation would be useful in further enhancing the detection for underrepresented or overlapping classes. In general, the model is excellent at multi-class network traffic classification, and it is making confident, solid predictions across the dataset with a proposed narrow margin of improvement in terms of handling minority or confused classes. Breaking down class-wise allows for a more sensitive appreciation of the model's performance beyond overall accuracy, acknowledging areas for improvement as well as improved-balanced class recognition.

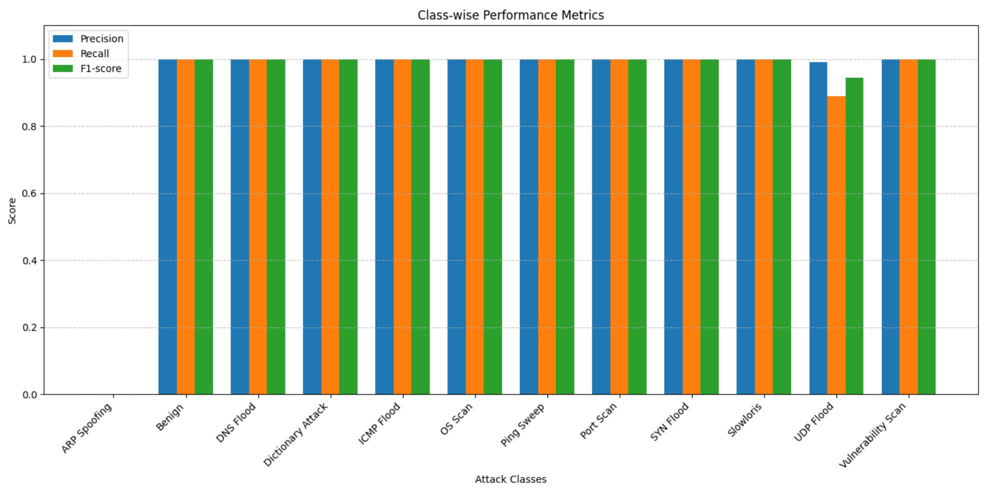

Fig. 4: Class-wise Performance Analysis

E. *Confusion Matrix Insights*

The confusion matrix in Figure 5 provides an in-depth view of the accuracy with which the proposed model classifies IoT traffic of different classes based on the ACI-IoT-2023 dataset. The dataset comprises 1,231,411 flow records, with 80% training and 20% test data, including both benign and different types of attacks. The result shows that the model achieves excellent high-classification accuracy in almost all classes. For instance, 65,859 benign flows, 9,387 DNS Flood flows, 45,047 ICMP Flood flows, 7,505 OS Scan flows, 14,386 Ping Sweep flows, and 88,257 Port Scan flows were accurately classified with very minimal misclassifications. Similarly, dictionary attacks, SYN floods, Slowloris, UDP floods, and vulnerability scans were also accurately labeled as seen in the high diagonal dominance of the matrix.

An extremely low number of misclassifications were reported. Specifically, one benign instance was mislabeled as ARP Spoofing, and 17 UDP Flood instances were mislabeled as ARP Spoofing. These low counts of misclassifications are likely because there is overlap among flow-level statistical features between low-volume ARP Spoofing and certain benign or UDP flood traffic, where packet size and frequency distributions share similar signatures. But these errors are very few in terms of the size of the dataset as a whole, indicating the strength of the model. Diagonal dominance of the matrix ensures that the model can also distinguish between various categories of IoT attacks and normal traffic very accurately. High-accuracy attacking classification is even more crucial for smart cities developed on IoT, with heterogeneous devices creating varied traffic patterns. The confusion matrix also illustrates that not only is the model distinguishing volumetric attacks like ICMP Flood and Port Scans, but also distinguishing low-rate stealthy attacks like Slowloris and dictionary-based intrusions. In summary, the confusion matrix shows the suggested model distinguishes almost perfectly, with very few margin misclassifications within overlapping statistical pattern groups. This ability largely enables real-time applicability of the model in actual IoT networks for real-time intrusion detection and prevention.

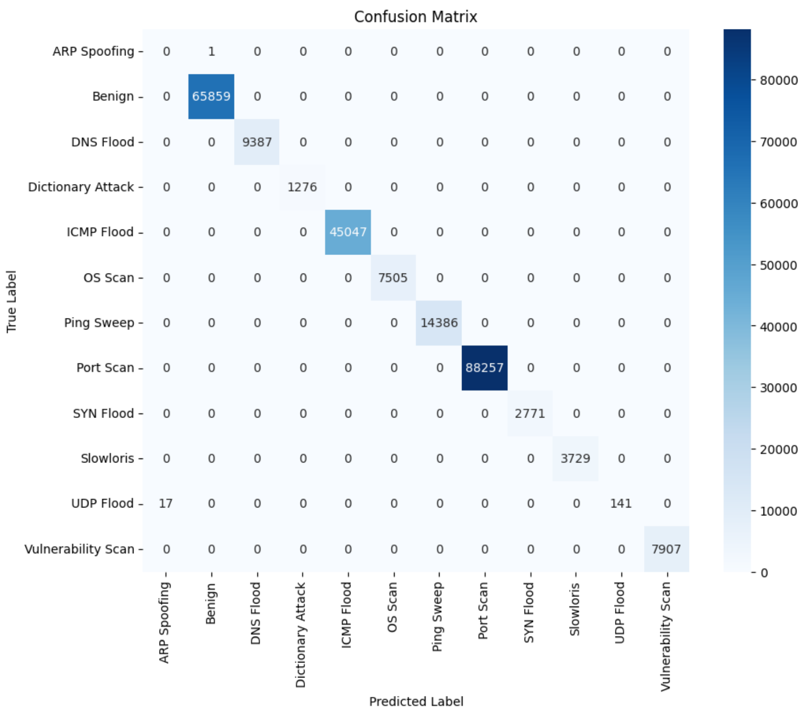

Fig. 5: Confusion Matrix for the proposed Model

F. Training Dynamics and Convergence Behavior Analysis

- Training Accuracy and Validation Accuracy Analysis

The Figure 6 accuracy plots display the learning curve of the proposed model. Training accuracy begins at approximately 75% during the first epoch and maintains a steady increase throughout training. The accuracy goes beyond 95% by epoch 10, and after epoch 20, the curve converges tightly to 99.9%, finally reaching a constant value of 99.99% towards the final epochs. This steady enhancement reflects proper feature extraction and optimization while training. Similarly, validation accuracy follows the same trajectory as training accuracy, starting slightly below 75% and rapidly improving in parallel with the training curve. Validation accuracy is above 97% by epoch 12, and from epoch 20 onwards, it remains consistently above 99.5% with exceedingly minor deviations. The close agreement between training and validation curves is an indication that the model has excellent generalization capability, with neither overfitting nor underfitting. The tight intersection of training and validation accuracies at 99.99% illustrates the power of the suggested model to handle complex IoT traffic patterns and distinguish between benign and malicious flows with outstanding accuracy. Achieving such a high level of accuracy is particularly significant for intrusion detection in IoT-enabled networks, where even minimal misclassifications can have severe consequences for security breaches.

- Training loss and Validation Loss Analysis

The training and validation loss curves shown in Figure 6 provide insight into the learning pattern of the presented model. The blue curve that indicates the training loss shows a highly steep decline at initial epochs, from approximately 0.48 to 0.15 in the first 5 epochs. The sudden drop indicates intensive learning and proper weight optimization during the initial phases of training. Past epoch 10, the loss in training converges and stabilizes at 0.11–0.12, which shows consistent model performance without overt overfitting. The validation loss also exhibits similar behaviour, being around 0.12 to begin with and progressively decreasing throughout training. There is a slight wobble in the vicinity of epoch 20, when a dramatic spike occurs, potentially a consequence of transient instability in gradient updates or batch-level fluctuations. The model is, nevertheless, quickly back on track, with the validation loss resuming its decreasing trend, reaching a minimum of 0.0485, as pointed out in the figure.

The general trend of the two curves demonstrates that the model generalizes very well, as the validation loss is always lower than the training loss, demonstrating excellent regularization and absolutely no evidence of overfitting. The tiny oscillations in the validation curve are expected in real-world data due to fluctuations in traffic patterns. In conclusion, loss curve analysis ensures that the model suggested has quick convergence, optimization stability, and fine generalization performance, which enables it to be applicable for handling large-scale IoT intrusion detection tasks.

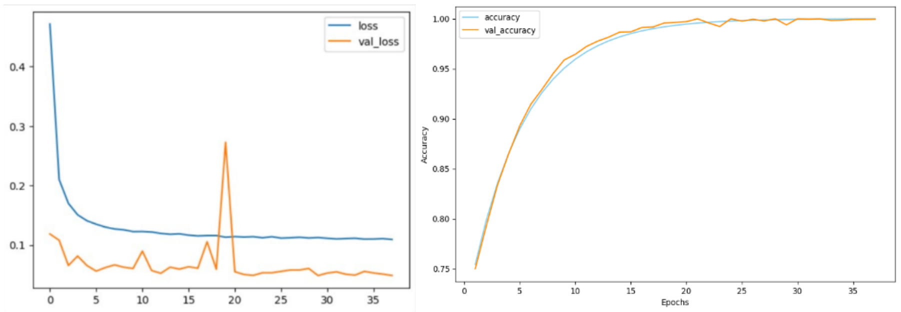

Fig. 6: Model's loss and accuracy

G. *ROC curve analysis*

The ROC curve depicted in Figure 7 provides a comparative evaluation of different classification models in terms of their discriminative power. The graph is a plot of the True Positive Rate against the False Positive Rate for LR, SVM, RF, DT, and the proposed hybrid DL model, Convolutional Neural Network with Long Short-Term Memory (CNN+LSTM). As it is shown, the CNN+LSTM model outperforms all the baselines significantly, achieving an Area Under the Curve (AUC) of 0.999. The provided value is that of the micro-average AUC, where predictions are micro-averaged over all 246,283 test samples in a one-vs-rest approach. The result is in line with the discovered overall accuracy of 99.99%, demonstrating the capability of the model to achieve near-perfect separation for every traffic category. For completeness, macro-average AUC and weighted-average AUC were also calculated, both of which were more than 0.999, indicating that the model still has consistent separability for both majority and minority classes.

Conversely, classical ML models performed comparatively poorly. Among these, RF had the highest discriminative ability (AUC = 0.943), followed by SVM (AUC = 0.931) and DT (AUC = 0.925). LR achieved an AUC of 0.912 with high classification ability, but less than in the ensemble-based and sequential ones. Critically, in comparing individual DL models by themselves, CNN and LSTM both outperformed standard ML baselines. CNN conducted an independent AUC of 0.981, and that reflects its capability in learning spatial patterns from flow-level features. In the same vein, LSTM attained an independent AUC of 0.987, which reflects its capability in learning temporal dependencies between sequential traffic flows. While the two models were quite good separately, when combined in the CNN+LSTM hybrid, there was a synergistic relationship, with great discrimination at AUC = 0.999. The consistent placement of the CNN+LSTM ROC curve above everything else reflects the capacity of deep sequential models in detecting complex, non-linear relationships between IoT traffic patterns. While the performance of conventional models is good, they are not sufficient to recognize temporal relationships and complex interactions between features. The above comparison highlights the importance of model selection in IoT intrusion detection, where DL models demonstrate a stark advantage with regard to near-optimal classification performance.

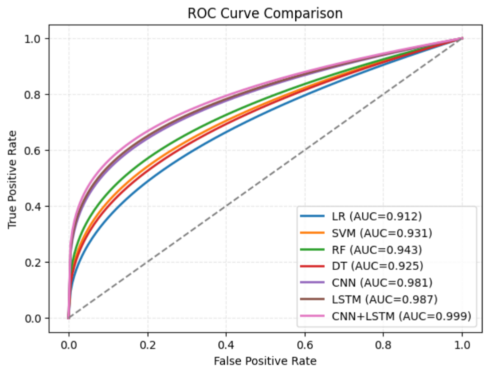

Fig. 7: ROC curve of the proposed model with baselines

V. CONCLUSION AND FUTURE WORK

The growing complexity of IoT networks, especially of smart cities, renders them a highly desirable target for cyber-attacks. Herein, in this paper, we were trying to develop an intrusion detection system capable of beating traditional rule-based and surface ML approaches. We wanted to build something that not only classifies the attacks efficiently but also learns to adapt to the evolving traffic patterns of IoT systems. In order to do this, we took into account a hybrid DL model that integrates CNN layers to learn features and LSTM layers to learn sequences. When the model was tested on the ACI-IoT-2023 dataset, it performed significantly better than a large repository of baseline methods with almost perfect accuracy. These results show that there is a lucrative future for IoT network security through the integration of spatial and temporal learning. All the same, the work is not perfect. The model also misidentified some attacks sometimes, for example, confusing UDP Flood with ARP Spoofing, showing that high-level traffic similarities may also be difficult to catch. Much more importantly, the experiment only used a single dataset. Real-world IoT deployments are significantly more heterogeneous, and cross-validation across multiple datasets must be conducted before conclusions can be reached more definitively. Future work will focus on minimizing the overhead of architecture and deployment on resource-constrained IoT devices, exploring online learning methodologies for real-time adaptation to shifting attack patterns, and the incorporation of explainable AI in an effort to provide network administrators with effective and actionable intelligence. These steps will help to evolve the proposed model to a reliable and viable intrusion detection system.

REFERENCES

1. Raza, M. S., Nowsin, M., Sheikh, A., & Hwang, I.-S. (2025). Ensemble learning-based DDoS attack recognition in IoT networks. *Computer Networks and Communications*, 73–83. https://doi.org/10.37256/CNC.3220256755
2. Al Dawi, A., Tezel, N. S., Rahebi, J., & Akbas, A. (2025). An approach to botnet attacks in the fog computing layer and Apache Spark for smart cities. *Journal of Supercomputing, 81*(4), 1–30. https://doi.org/10.1007/s11227-024-06915-y
3. Chennupati, N., Gottam, J., Marrelli, R., & Panda, A. (2025). Enhancing IoT intrusion detection with Greylag Goose Optimization and Extreme Learning Machine: A data-driven study on IoT23. *ResearchGate*. https://www.researchgate.net/publication/394046831_Enhancing_IoT_Intrusion_Detection_with_Greylag_Goose_Optimization_and_Extreme_Learning_Machine_A_Data-Driven_Study_on_IoT23
4. Aloqaily, A., Abdallah, E. E., AbuZaid, H., Abdallah, A. E., & Al-Hassan, M. (2025). Supervised machine learning for real-time intrusion attack detection in connected and autonomous vehicles: A security paradigm shift. *Informatics, 12*(1), 4. https://doi.org/10.3390/informatics12010004
5. Shan, L. (2025). IoT network intrusion detection system using optimization algorithms. *Scientific Reports, 15*(1), 1–19. https://doi.org/10.1038/s41598-025-04638-5
6. Rahmani, A. M., et al. (2022). A particle swarm optimization and deep learning approach for intrusion detection system in Internet of Medical Things. *Sustainability, 14*(19), 12828. https://doi.org/10.3390/su141912828
7. Lazrek, G., Chetioui, K., Balboul, Y., Mazer, S., & El Bekkali, M. (2024). An RFE/Ridge-ML/DL based anomaly intrusion detection approach for securing IoMT system. *Results in Engineering, 23*, 102659. https://doi.org/10.1016/j.rineng.2024.102659
8. Urs, P. M., Reddy, A. T. N., Mallikarjunaswamy, S., & Lakshminarayan, U. M. (2025). An innovative IoT framework using machine learning for predicting information loss at the data link layer in smart networks. *Engineering, Technology & Applied Science Research, 15*(2), 20904–20911. https://doi.org/10.48084/ETASR.9597
9. Ahmed, Y., Beyioku, K., & Yousefi, M. (2024). Securing smart cities through machine learning: A honeypot-driven approach to attack detection in Internet of Things ecosystems. *IET Smart Cities, 6*(3), 180–198. https://doi.org/10.1049/smc2.12084
10. Abbasi, M., Shahraki, A., Prieto, J., Arrieta, A. G., & Corchado, J. M. (2024). Unleashing the potential of knowledge distillation for IoT traffic classification. *IEEE Transactions on Machine Learning in Communications and Networking, 2*, 221–239. https://doi.org/10.1109/TMLCN.2024.3360915
11. Liao, N., & Guan, J. (2024). Multi-scale convolutional feature fusion network based on attention mechanism for IoT traffic classification. *International Journal of Computational Intelligence Systems, 17*(1), 1–25. https://doi.org/10.1007/s44196-024-00421-y
12. Afifi, F., Zaki, F., Hanif, H., Aqil, N., & Anuar, N. B. (2025). Transformer-based tokenization for IoT traffic classification across diverse network environments. *PeerJ Computer Science, 11*, e3126. https://doi.org/10.7717/peerj-cs.3126
13. Ismail, S., Dandan, S., & Qushou, A. (2025). Intrusion detection in IoT and IIoT: Comparing lightweight machine learning techniques using TON_IoT, WUSTL-IIoT-2021, and EdgeIIoTset datasets. *IEEE Access, 13*, 73468–73485. https://doi.org/10.1109/ACCESS.2025.3554083
14. Zahid, M., & Bharati, T. S. (2025). Enhancing cybersecurity in IoT systems: A hybrid deep learning approach for real-time attack detection. *Discover Internet of Things, 5*(1), 1–31. https://doi.org/10.1007/s43926-025-00156-y
15. Miao, Z., & Liao, Q. (2025). IoT-based traffic prediction for smart cities. *IEEE Access, 13*, 52369–52384. https://doi.org/10.1109/ACCESS.2025.3552276

16. An, R., Zhang, X., Sun, M., & Wang, G. (2024). GC-YOLOv9: Innovative smart city traffic monitoring solution. *Alexandria Engineering Journal, 106*, 277–287. https://doi.org/10.1016/j.aej.2024.07.004
17. Nack, E. A., McKenzie, M. C., & Bastian, N. D. (2024). ACI-IoT-2023: A robust dataset for Internet of Things network security analysis. In *Proceedings of the IEEE Military Communications Conference (MILCOM)*. https://doi.org/10.1109/MILCOM61039.2024.10773916
18. Yang, J., et al. (2025). BrainCNN: Automated brain tumor grading from magnetic resonance images using a convolutional neural network-based customized model. *SLAS Technology, 34*, 100334. https://doi.org/10.1016/j.slast.2025.100334
19. Ali, M., et al. (2025). Improving daily reference evapotranspiration forecasts: Designing AI-enabled recurrent neural networks based long short-term memory. *Ecological Informatics, 85*, 102995. https://doi.org/10.1016/j.ecoinf.2025.102995
20. Zhang, J., Lai, Z., Kong, H., & Yang, J. (2025). Learning the optimal discriminant SVM with feature extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 47*(4), 2897–2911. https://doi.org/10.1109/TPAMI.2025.3529711
21. Wu, W. (2025). Research on customer traffic value recognition model based on improved random forest algorithm. *International Journal of Business, Management and Economics Technology*. https://doi.org/10.38007/IJBMET.2025.060109
22. Achari, A. P. S. K., & Sugumar, R. (2025). Performance analysis and determination of accuracy using machine learning techniques for decision tree and RNN. *AIP Conference Proceedings, 3252*(1). https://doi.org/10.1063/5.0258588
23. Sinha, P., Sahu, D., Prakash, S., Yang, T., Rathore, R. S., & Pandey, V. K. (2025). A high performance hybrid LSTM-CNN secure architecture for IoT environments using deep learning. *Scientific Reports, 15*(1), 1–26. https://doi.org/10.1038/s41598-025-94500-5
24. Pasha, A., Ahmed, S. T., Painam, R. K., Mathivanan, S. K., Mallik, S., & Qin, H. (2024). Leveraging ANFIS with Adam and PSO optimizers for Parkinson's disease. Heliyon, 10(9).
25. Ahmed, S. T., Priyanka, H. K., Attar, S., & Patted, A. (2017, June). Cataract density ratio analysis under color image processing approach. In 2017 International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 178-180). IEEE
26. Ahmed, S. T., Kumar, V. V., & Jeong, J. (2024). Heterogeneous workload-based consumer resource recommendation model for smart cities: EHealth edge–cloud connectivity using federated split learning. *IEEE Transactions on Consumer Electronics*, *70*(1), 4187-4196.