TCH-NET: Temporal-Contextual Hybrid Network for IoT Botnet Mitigation and Prevention

Kiran Tomar, Vikash Thada, and Umang Garg*

Original scientific article

***Abstract*—The fastest growing Internet of Things (IoT) has greatly raised the attack space of contemporary networks, allowing the development of large-scale and highly adaptable IoT botnets. The modern botnets take advantage of device heterogeneity, insecure authentication protocols, and dynamic command-and-control (C2) systems, which makes all the existing signature-based and independent machine learning methods of detection ineffective. Most of the existing solutions cannot simultaneously capture the temporal evolution of attacks and the contextual behavioral attributes of IoT traffic, which makes them highly vulnerable to false alarm and delayed response despite the recent deep learning (DL) models showing better detection rates. To address this gap, the key novelty of this work lies in the design of TCH-NET, a unified temporal–contextual hybrid framework that jointly models attack progression patterns and device-level behavioral semantics, along with an integrated real-time mitigation and prevention mechanism. The proposed framework integrates a temporal learning component to learn sequential behavior of traffic and transitions between attack stages with a contextual feature learning component to learn flow-level statistics, protocol semantics, and device abnormal behavioral patterns. To identify strong discrimination against normal and malicious activities, a hybrid feature fusion mechanism is used to merge temporal dependencies and contextual intelligence. The model includes an adaptive mitigation and prevention layer, which facilitates real-time threat scoring, automated response and dynamic enforcement of security policies. A large-scale test is performed on benchmark N-BaIoT dataset to assess the efficiency of the proposed framework. The experimental outcome shows the high performance of the proposed TCH-NET model in terms of accuracy, F1-score, false alarm rate, and early attack detection.**

***Index Terms*—IoT Botnet, Malware Classification, behavioral pattern, Hybrid Network.**

I. Introduction

IoT has become a base technology, which allows intelligent environments, including transportation systems, health care, industrial automation, grids, and smart cities. IoT systems support real-time data capture, automation and decision-making by linking together billions of heterogeneous devices such as sensors, actuators, embedded controllers, and smart appliances. Recent incidents suggest that the count of interconnected IoT devices will grow beyond several tens of billions in the next decade resulting in an unprecedented increase in the size and complexity of networks [1]. The majority of IoT devices are resource-constrained by nature about processing power, memory, storage and energy availability. Therefore, they often do not have sophisticated cryptography, secure booting, and patching. Figure 1 shows the botnet threat and distinct phases that required an effective framework to provide the end-end to security. The framework is created by modeling the temporal traffic evolution together with the contextual behavioral semantics in a single DL system. TCH-NET integrates both dimensions to recognize different threats posed by advanced IoT botnets early, classify them correctly, and mitigate them in advance.

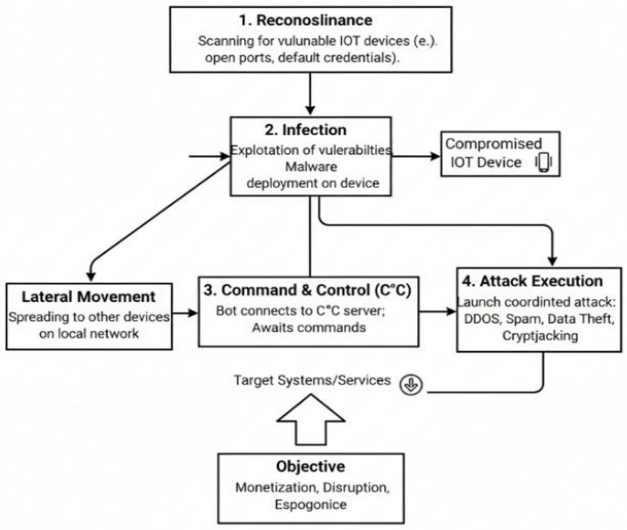


Fig. 1. IoT Botnet Threat

The conventional centralized security frameworks that are based on cloud intrusion detection and traffic analysis cannot be effectively scaled in large-scale IoT environment as they have significant communication overhead, latency, and have single points of failure [2]. A discrepancy between protection demands and the capabilities of the devices provides a very good soil to be exploited by multitudes, especially in the shape of IoT botnets. IoT botnet has gained popularity in the level of sophistication, magnitude, and the ability to attack. The disastrous consequences of the use of default credentials in poorly secured IoT devices were demonstrated by the record-

Manuscript received March 16, 2026; revised March 25, 2026. Date of publication September 15, 2026. Date of current version September 15, 2026. The associate editor prof. Toni Perković has been coordinating the review of this manuscript and approved it for publication.

K. Tomar and V. Thada are with the Department of Computer Science and Engineering, Amity School of Engineering and Technology, Gwalior, Madhya Pradesh, India (e-mails: kiran.tomar87@gmail.com, vthada@gwa.amity.edu).
U. Gang is with the School of Computer Science and Engineering, IILM University, Gurugram, India (e-mail: umangarg@gmail.com).

Digital Object Identifier (DOI): 10.24138/jcomss-2026-0060

*Corresponding author

breaking distributed denial-of-service (DDoS) attacks by early botnets like Mirai. The variants of Mirai derived continued with many others, such as Bashlite, Hajime, and Satori, each adding better propagation policies and buildings composed of modules. IoT botnets such as Mozi, Dark Nexus and other peer-to-peer (P2P) botnets now maintain sophisticated capabilities such as decentralized C2 communication, peer discovery that is dynamic, and encrypted transmissions of payload [3].

Recent IoT botnets use obfuscated and polymorphic behaviors, which allows attack signatures to be mutated many times to avoid detection systems. Contemporary botnets operate in a multi-vector mode, such as volumetric DDoS flooding, network scanning, brute-force attacks with credentials, lateral movement, and exfiltration of data as opposed to single-vector attacks [4]. These attack campaigns are usually implemented over a period and would occur in.

To mitigate the threat of IoT botnets, many detection tools have been suggested, including signature-based intrusion detection systems (IDS) as well as ML and DL models. These techniques have some issues for the identification and detection of botnets. The signature-based IDS solutions are based on pre-established signature patterns and ruleset, and thus cannot respond to zero-day attacks, encrypted traffic and new variants of botnets. These systems take time to update manually and cannot extrapolate under different IoT systems.

Despite the improved detection accuracy of deep learning models like convolutional neural networks (CNNs) and recurrent neural networks (RNNs), most of the current solutions are based on the individual use of spatial and temporal features. There is a lack of contextual understanding including device behavior profiles, protocol semantics and relationships at the flow level, which in turn decreases the reliability of detection and adds to high-rate false-positives [5]. The contextual features like packet statistics, protocol usage and device identity are considered useful in getting behavior insight, but they cannot capture the long-term dependencies and attack-stage transitions. Traffic movements are divided into three severity levels based on the calculated threat confidence level and the type of attack detected:

- Low severity: The suspicious deviations that possess a low confidence level and are frequently found in the initial scanning efforts.
- Medium severity: Vindicated botnet-related communication like C2 beaconing.
- High severity: Active implementation of the attack involving DDoS flooding or mass scanning.

The classification of severity enables the system to balance response activities and prevent unwanted interference of legitimate IoT activities. After identifying malicious activity, the mitigation module activates automatic mitigation measures to reduce the effects of attacks without affecting the normal services. In situations where blocking in real-time can have service continuity implications, a rate limit is implemented as a soft mitigation measure [6]. Detection of repeated malicious behavior incurs the placement of compromised IoT devices in a quarantine network segment.

The prevention mechanism dynamically sets up firewall rules based on the attack patterns [7]. The rules are self-learning from the new patterns of attacks of the botnets, such as unusual port numbers, excessive number of connection attempts, and unusual protocol patterns. Limited DNS query patterns can generate tighter DNS policies. Although correct identification is a critical prerequisite, adequate protection against the IoT botnets demands prompt mitigation and preventive mechanisms in the long term. Thus, the suggested TCH-NET structure incorporates detection, mitigation, and prevention in a single security architecture. When suspicious or malicious traffic is detected, real-time alerts are generated by the TCH-NET detection module to give actionable intelligence data to the network administrators and automated security systems. TCH-NET does not generate binary decisions only but calculates the confidence score of a threat on each traffic flow being monitored. This score is based on the result of the classification probability of the hybrid learning model and whose value measures the probability of a given flow to be part of a specific class of botnet activities [8].

One of the most important innovations of the TCH-NET framework is feedback learning loop that connects between detection and prevention. The mitigation and prevention mechanisms are distributed on a distributed edge -fog -cloud architecture to facilitate scalability and real-time response. Primary traffic monitoring and initial TCH-NET inference are conducted at edge nodes [9]. The framework allows to rapidly contain the IoT botnet threats with minimal operational disruption through real-time alert generation, automated response plans, adaptive prevention policy, and distributed deployment. An intelligent-based detection model that combines both temporal learning and contextual awareness is needed to provide relevant, resilient, and scalable defense against botnet IoT.

The major contributions of the current article are as follows:

- To develop a temporal contextual hybrid detection model that would be able to simulate both the dynamics of IoT traffic and semantics of its behavior.
- The improvement will be made to detect botnets early on by detecting soft attack transitions, where massive exploitation is yet to happen.
- To build an adaptive mitigation and prevention mechanism that would facilitate real-time response and dynamism in enforcing security policies.

The current article proposes a smart solution to the IoT botnet detection, mitigation, and prevention by designing a new Temporal-Contextual Hybrid Network (TCH-NET) architecture. The hybrid architecture learns time-dependent traffic patterns and contextual behavioral features simultaneously. The remainder of the paper is structured in the following sections: Section II discusses literature work conducted by the researchers. The proposed TCH-NET model is discussed in Section III. Section IV covers the dataset description with the categorization of dataset. Experimental setup and configuration of system are explained in Section V. The results and explanation of results are discussed in Section VI. Finally, Section VII concludes the article with future scope.

II. Literature Review

In recent traditional methods of detecting botnets depend on signature-based IDS and rule-based systems that compare network traffic with known signature threats or behavioral rules. The nature of signature-based IDS is also restrictive in that they are incapable of identifying zero day or polymorphic variants of botnet that do not match the set signature definitions [10]. These systems have encrypted messages and changing botnet patterns, which lead to low detection rates and large false alarms in more complicated IoT systems. The rule-based systems apply fixed heuristics of known malicious practices and are not effective against dynamic and stealthy botnets that actively attempt to obfuscate behavior.

The ensemble ML systems that consist of RF, XGBoost, and LightGBM have been tested on the IoT botnet detection in the edge-computing setting and found that these ML models can detect the dynamic botnet traffic patterns with encouraging accuracy and could be deployed to resource-intensive edge devices [11]. RF, multi-layer perception (MLP), SVM, and naive Bayes demonstrated that RF could provide an overall high detection rate 95% when classifying botnet traffic, and that ensemble ML strategies can help to capture non-linear relationships between network flow data. An additional hybrid ML model stacked RNN, CNN, ANN and classical ML models indicating a high level of generalizability to different types of attacks on the UNSW-NB15 data set [12]. This study demonstrates the ability of diversified ML pipelines to enhance the capture of advanced botnet traffic signatures. ML models are generally vulnerable to sensitive feature engineering, are sensitive to class imbalance, and do not represent the time well [13].

TABLE I
Literature Review

Author(s) & Reference	Contribution	Outcome	Limitation
Alkahtani et al. [23]	Introduced real-world labeled IoT botnet dataset	Enabled benchmarking of botnet detection models	No detection or mitigation framework proposed
Hajla et al. [24]	Behavioral analysis of Mirai botnet traffic	Identified attack signatures and traffic patterns	Signature-based; ineffective for evolving botnets
Ferrag et al. [25]	Large-scale synthetic IoT botnet dataset	Facilitated ML/DL evaluation	Synthetic traffic lacks real-world diversity
Islam et al., [26]	Survey of IoT intrusion detection	Comprehensive taxonomy of IDS methods	No experimental validation
Vinayakumar et al., [28]	Deep learning IDS using DNN	Improved detection over ML methods	High training cost; lacks temporal modeling
Nguyen et al., [6]	CNN-based botnet detection	High spatial feature learning capability	Ignores temporal dependencies
Alothman et al., [27]	LSTM-based temporal IDS	Captured sequential attack behavior	High false positives in dynamic traffic
Popoola et al., [9]	GAN-based IoT botnet detection	Addressed data imbalance	Training instability and complexity
Moorthy et al., [29]	Attention-based CNN-LSTM	Enhanced feature focus	Computationally expensive
Liang et al., [30]	Ensemble ML approach	Reduced variance and overfitting	Poor temporal modeling
Manasrah et al., [31]	Deep ensemble IDS	Improved robustness	High inference latency
Nguyen et al., [7]	BiLSTM-based IoT IDS	Long-term dependency learning	Ignores flow-level semantics
Putra et al., [32]	Graph-based IoT botnet detection	Modeled device communication topology	Scalability issues in large networks
Wang et al., [14]	GNN with attention mechanism	Captured inter-device relationships	High memory and processing overhead
Xing et al., [33]	Federated learning-based IDS	Preserved data privacy	Detection accuracy is lower than centralized models
Joshi et al., [34]	Edge-based lightweight DL IDS	Suitable for resource-constrained IoT	Reduced accuracy under complex attacks
Research Gap Analysis	—	—	Lack of unified temporal–contextual–mitigation framework

The hybrid CNN-LSTM models can learn both the spatial patterns and time dynamics and provide correct detection on internet-of-things data, like N-BaIoT [14]. RNN and LSTM models have been studied extensively in terms of learning temporal patterns, and some of these studies have reached a very high level of accuracy in catching botnet and malware traffic, by learning sequential characteristics of packet streams, and temporal relations. Studies on SDN-based IoT networks practical specifically state that LSTM-based models have a 99% detection rate when it comes to botnet detection using their long-range temporal correlation capabilities [15].

Attention-based CNN-BiLSTM models aimed at increasing feature and temporal encoding have also been studied recently. These models have shown classification performance 99% on benchmark botnet data sets, confirming the usefulness of the attention mechanisms to highlight useful temporal information in the complex internet traffic [16]. The objectives of these models are to learn normal distributions of the IoT traffic and raise an alarm when the traffic deviates, which is important in real-life situations that might have limited labeled data. They tend to need a lot of training information and cannot easily adapt to concept change in network traffic [17]. Indicatively, hybrid CNN-LSTM models have been suggested with the aim of utilizing the spatial feature of CNN with the temporal characteristics of LSTM learning, making hybrid models that can identify more complex attack sequences in botnets with enhanced accuracy [18]. The research [19] presents hybrid deep learning models based on LSTM Autoencoders with multilayer perception, with a detection rate of almost 99.77% on N-BaIoT and UNSW-NB15. These hybrid models alleviate the few limitations of standalone DL approaches as they provide an effective boost of both temporal and non-temporal learning of features. The hybrid frameworks which utilize attention mechanisms on the top of CNN-BiLSTM architectures have been observed to be more effective as it completely focuses the model attention on salient features [20].

Context-aware models take into consideration extra information besides the properties of packets to enhance classification. These are flow-based contextual features (e.g. protocol usage patterns, frequency of connection), device behavior profiles, and network topology information. Recent literature delves into GNN-based methods that describe the interactions between IoT traffic and devices as graphs allowing models to take advantage of structural patterns in network communication [21]. Graph Attention Networks (GAT) and heterogeneous GNNs have been introduced that can represent flow-level as well as temporal relationships among the devices and have demonstrated improved detection capabilities because of their capacity to learn topology and behavior simultaneously [22]. These research gaps that drive the necessity of hybrid paradigm whereby the temporal behavior, context-aware models, and actionable remedies are combined in a single framework which is the point of the proposed TCH-NET. Table I shows the recent literature with contribution, outcome, and limitation.

III. Proposed TCH-NET Framework

The proposed TCH-NET architecture is based on a layered architecture that consists of four significant functional elements, namely, data acquisition, feature extraction, hybrid learning, and decision and mitigation (Figure 2).

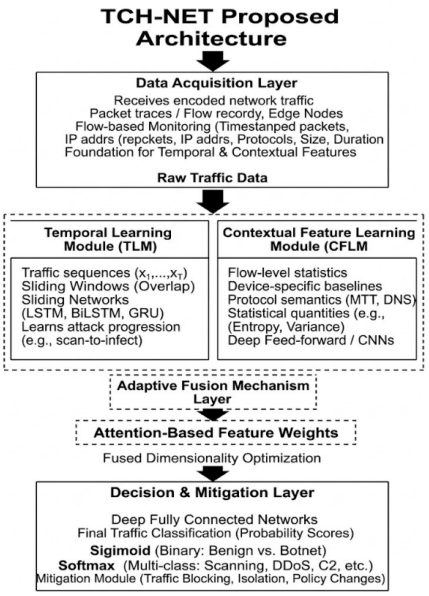


Fig. 2. Workflow of the Proposed Approach

Data Acquisition Layer: This layer overseas receives encoded network traffic through distributed IoT settings. Packet traces or flow records represent traffic recorded at the gateway, edge nodes or network taps. The framework is based on flow-based monitoring as opposed to deep packet inspection to be able to achieve scalability and preserve privacy. The information gathered can be by way of timestamped packets, source and destination addresses, port numbers, types of protocols, size of packet, duration of flow and statistics of connection. These records are the foundation of the construction of the temporal sequence as well as the extraction of the features of context.

Feature Extraction Layer: The feature extraction layer converts raw traffic to structured representations used in deep learning. There are two types of features derived from temporal features, which describe sequential dependencies and the temporal change of traffic and contextual characteristics, which are behavioral and semantic aspects of the IoT communications. This isolation allows complementary information to be independently learnt before hybrid fusion.

Hybrid Learning Layer: The TCH-NET mainly consists of the hybrid layer of learning. It is composed of two parallel learning modules, temporal learning module (TLM), and contextual feature learning module (CFLM). Each of the modules works with different sets of features and learns representation specific to it. They then generate their results and fuse together with an adaptive fusion mechanism to create a hybrid representation.

Decision and Mitigation Layer: Final traffic classification is done at the decision layer via deep fully connected networks whereby probability scores of the benign and various categories of botnet attacks are generated. Depending on the estimated threat intensity, the mitigation module initiates the relevant response measures like traffic blocking, device isolation or policy changes.

A. TCH-NET Design

TCH-NET has an architecture with three principles of design (Algorithm TCH-NET). The IoT botnet operations are progressive through various phases such as reconnaissance, infection, coordination and execution of attack. Modeling long-term temporal dependencies and capturing such progression involves capturing long-term flow dependencies as opposed to capturing flow snapshots. Intelligence Temporal intelligence allows the detection of minor attacks at the early stages before major attacks are caused. The presence of device identity, protocol semantics and flow relationships help a lot in improving detectability. Contextual and temporal characteristics offer two complementary views of the behavior of networks. Hybrid fusion has allowed the model to attribute sequential anomalies to behavioral context and decrease ambiguity and enhance generalization against unknown variants of botnets.

Algorithm TCH-NET
Input: IoT traffic dataset $D = \{(x_i, y_i)\}$, Temporal window size w, Model parameters θ, parameter classes x.
Output: Predicted label $\hat{y}$, Threat score S, Mitigation action A.
Step 1: Data Acquisition: Collect network traffic flows from IoT devices through gateways or edge nodes.
Step 2: Data Preprocessing:
- Remove noise and invalid records
- Handle missing values
- Normalize features using Min-Max scaling

Step 3: Temporal Sequence Construction
- Sort traffic data by timestamp
- Create sequences using sliding window (S_t) of size w:

$$S_t = \{x_{t-w+1}, \ldots, x_t\}$$

Step 4: Temporal Feature Learning
- Input sequences into BiLSTM/GRU
- Extract temporal representation $h^{(T)}$

Step 5: Contextual Feature Extraction
- Extract statistical, behavioral, and protocol features
- Pass through dense layers to obtain $h^{(C)}$

Step 6: Hybrid Feature Fusion
- Compute attention weights
- Combine features:

$$H = \alpha_T h^{(T)} + \alpha_C h^{(C)}$$

Step 7: Classification
- Apply fully connected layers
- Use Softmax/Sigmoid to predict class:

$$\hat{y} = \arg\max P(y \mid x)$$

Step 8: Threat Scoring
- Compute confidence score:

$$S = \max(P(y \mid x))$$

Step 9: Mitigation Decision
If $S > \theta$, apply: Flow blocking, IP blacklisting, Rate limiting, Device quarantine.
Step 10: Model Training
- Compute loss (cross-entropy)
- Update parameters using backpropagation

Step 11: Feedback Learning
- Store detected attacks
- Update model periodically for improved performance

Temporal Learning Module (TLM): The TLM is implemented to learn the sequence of traffic behavior and evolution pattern of attacks. The network flows are arranged in time-sequences with fixed or flexible time windows. Each sequence is an expression of the behavior of the traffic during a continuous observation time. This representation does not discard inter-flow dependencies that are destroyed in the traditional feature-based classifiers. Let X = {x_1, x_2, …., x_n} represent a traffic sequence of size n, in which the elements have temporal flow attributes.

TCH-NET allows several recurring structures such as LSTM on learning long-range dependencies, BiLSTM to learn forward and backward time relations, computational efficiency gated recurrent unit (GRU). These models are trained to learn representations of hidden states that encode temporal dependencies between traffic sequences. The temporal module trains transitions between attacks, e.g. scanning-to-infection or beacon-to-flood.

Contextual Feature Learning Module: CFLM is a module that aims at extracting semantic and behavioral features of IoT traffic. Flow-level statistics like the number of packets, the ratio of bytes, the inter-arrival time, and the connection persistence give information on communication. The traffic of botnets tends to be repetitive or symmetric or burst unlike the normal traffic of the devices. CFLM model's device specific baselines such as normal communication endpoints, frequency and protocol application. Such semantics improve interpretability and enhance resistance to evasion using encryption. Statistical quantities such as entropy, variance and burst ratios are derived to obtain randomness and coordination behavior, which are often characteristic of botnet behavior. Deep feed-forward networks or convolutional layer are used to acquire non-linear relationships between contextual features. Early fusion is the combination of raw features prior to learning which can easily cause dimensional explosion. TCH-NET embraces late fusion to ensure modeling independence and enhance scalability.

Attention-Based Feature Weights: The relevance is dynamically converted into weights assigned to the temporal and contextual embeddings by an attention mechanism to emphasize temporal indicators in the case of shift of attack and contextual indicators in the circumstances of deviant behavior. The techniques of batch normalization and layer normalization stabilize training and speed up the convergence. Normalization also eliminates bias due to the uneven scales of features. The next step is dimensionality optimization. The dimensionality reduction layers reduce the fused representations to minimize the inference cost, and to deploy the model with lightweight at edge nodes. Equation (1) indicates the prioritize temporal signals.

$$\alpha_i = \frac{e^{w_i}}{\sum_j e^{w_j}} \tag{1}$$

where α is the output probability and w represents the raw score.

Classification Layer: The classification layer identifies fused features to the end prediction values. Learned representations are refined by several dense layers with nonlinear activation functions. To reduce the occurrence of overfitting, dropout regularization is used. Binary (benign vs botnet) classification is done by using sigmoid activation. The multi-class detection of attack types can be considered a scanning, DDoS, brute force, and C2 communication with the help of Softmax activation. Probability scores of each class are given by the final output vector, and this allows fine-grained attack awareness. The mitigation engine further uses these probabilities to identify the severity of the response. The presented TCH-NET framework presents a single temporal-contextual intelligence-based approach to the detection of IoT botnet. It is based on this approach that it is holistic to build resilience against emerging threat of IoT botnets based on robust detection, proactive mitigation, and resilience over time. Successful botnet detection is highly dependent on the quality of the input data and thus, it is important to take care of data preparation and feature engineering to obtain reliable and repeatable results.

The TCH-NET architecture can be defined in terms of algorithmic steps:

Input:

- Network traffic dataset: $D = \{(x_i, y_i)\}_{i=1}^{N}$
- Temporal window size: w
- Feature vector: $x_i \in \mathbb{R}^d$

Output:

- Predicted class label: $\hat{y}$
- Threat score: S
- Mitigation action: A

Step 1: Data Preprocessing

Normalize features using Min-Max scaling:

$$x_i' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{2}$$

where x_i represents the parameter value and x_i' normalized values.

Construct temporal sequences (S_t):

$$S_t = \{x'_{t-w+1}, x'_{t-w+2}, \ldots, x'_t\} \tag{3}$$

Step 2: TLM

Each sequence is passed through BiLSTM:

$$\overrightarrow{h_t} = \text{LSTM}(x_t, \overrightarrow{h_{t-1}}) \tag{4}$$

$$\overleftarrow{h_t} = \text{LSTM}(x_t, \overleftarrow{h_{t+1}}) \tag{5}$$

Final temporal representation ($h_t^{(T)}$):

$$h_t^{(T)} = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{6}$$

Step 3: CFLM

Contextual features processed via dense layers ($h^{(C)}$):

$$h^{(C)} = \sigma(W_c x_i' + b_c) \tag{7}$$

$$h^{(C)} = \text{ReLU}(W_2 h^{(C)} + b_2) \tag{8}$$

Step 4: Attention-Based Hybrid Fusion

Compute attention weights (α_T, α_C):

$$\alpha_T = \frac{e^{W_T h^{(T)}}}{e^{W_T h^{(T)}} + e^{W_C h^{(C)}}} \tag{9}$$

$$\alpha_C = \frac{e^{W_C h^{(C)}}}{e^{W_T h^{(T)}} + e^{W_C h^{(C)}}} \tag{10}$$

Fused representation (H0):

$$H = \alpha_T \cdot h^{(T)} + \alpha_C \cdot h^{(C)} \tag{11}$$

Step 5: Classification Layer (z)

$$z = W_f H + b_f$$

Softmax output (P):

$$P(y = k \mid x) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}} \tag{12}$$

Predicted class ($\hat{y}$):

$$\hat{y} = \arg\max_k P(y = k \mid x) \tag{13}$$

Step 6: Loss Function

Cross-entropy loss ($\mathcal{L}$):

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

Step 7: Threat Confidence Scoring (S):

$$S = \max\big(P(y \mid x)\big)$$

Step 8: Mitigation Strategy

If $S > \theta$:

$$A = \begin{cases} \text{Block Flow}, & S > 0.9 \\ \text{Rate Limit}, & 0.7 < S \leq 0.9 \\ \text{Monitor}, & S \leq 0.7 \end{cases} \tag{14}$$

Step 9: Model Optimization

Update parameters using Adam optimizer:

$$\theta = \theta - \eta \cdot \nabla_\theta \mathcal{L}$$

The joint model can be represented as:

$$H = f(h^{(T)}, h^{(C)}).$$

IV. Dataset Description

This section presents the description about the N-BaIoT dataset. There was a gap in the availability of realistic IoT botnet traffic data, however, N-BaIoT (Network-Based Analysis of IoT Traffic) dataset was proposed that contain the realistic IoT botnet traffic data. It has since been one of the most popular datasets used to test the efficiency of intrusion detection systems in IoT setups. The data is the actual network traffic of commercial IoT devices which were infected with the Mirai and Bashlite botnets. In contrast to synthetic datasets, N-BaIoT record the real-world behavior of devices in controlled infection conditions, and it is especially helpful to study the dynamics of botnet spreading and attacks. Nine

actual IoT devices were monitored to collect traffic and they included (Figure 3):

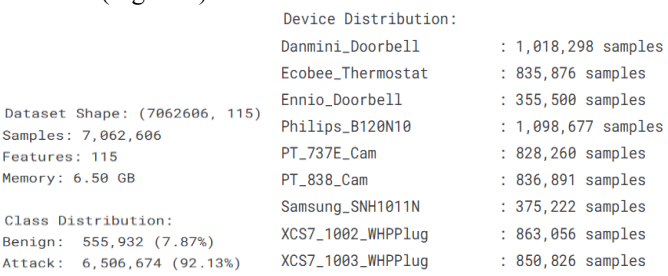


Fig. 3. Dataset sample and device distribution

The devices were also subjected to a realistic testbed and left to produce benign traffic before they were intentionally inoculated with botnet malware. The data is provided with traffic when various botnet attacks occurred such as TCP SYN flooding, UDP flooding, HTTP flooding, TCP ACK attacks, TCP scanning, and UDP scanning. Such attacks are typical actions of real-world IoT botnets and offer binary and multi-class classification opportunities. The N-BaIoT dataset has more than 7 million records of network flows, and each record is the aggregated statistics calculated across the time windows. The dataset, as opposed to raw packets, presents extracted features that save a lot of overhead in storage and processing. Every flow instance has 115 statistical features, which include packet count statistics, inter-arrival time metrics, packet size distributions, connection-based statistics, and directional traffic ratios.

A combination of these features tracks the short-term and long-term traffic behavior. Raw data usually has inconsistency problems which can negatively impact the running of the model. Hence, normalization and systematic data cleaning processes are used (Figure 4). The N-BaIoT data sources the noise is mostly due to incomplete flow records, radical artifact outliers, and features that are constant in nature and non-informative. Attributes that have zero variance across all the samples were removed, because they do not play a role in classification. The median imputation was used to fill in the numerical missing values. The median based imputation is used instead of the mean substitution to minimize sensitivity to skewed distributions typical of network traffic. There are large dynamic ranges of feature values in N-BaIoT. Min -max normalization was used to stabilize neural network training, it increases the speed of convergence, eliminates dominance of big features, and increases the ability to generalize.

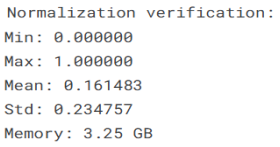


Fig. 4. Normalization Verification

To enable the suggested TCH-NET architecture, high-level feature engineering methods were utilized to create temporal and context-sensitive representations. N-BaIoT has flow-based capabilities that does not explicitly capture temporal relationships. Thus, there was reorganization of traffic records into time-series sequences.

The contextual features were computed to extract semantic data on top of raw statistics. These include direction ratio of flows, packet size entropy, protocol usage frequency, connection persistence measures, and request–response asymmetry. This allows the model to learn normal behavioral baselines of each type of device. The features are extracted based on the characteristics such as time, statistical, and behavioural. TCP/UDP distribution, and patterns of service access classified as semantic of protocol. This hierarchical presentation facilitates the process of learning in both temporal and contextual modules of TCH-NET. The data set shows a high level of class imbalance with the attack traffic prevailing over benign samples since the attack on the botnet is a continuous process. To address data imbalance, stratified sampling is utilized, training is done on minority benign samples, and use of weighted loss functions. The method helps to avoid the biasness in relation to the dominant classes of attacks and has realistic proportions of traffic. The processed data were divided 3 distinct categories 70% for training, 15% for validation, and 15% for testing. The N-BaIoT data set offers a realistic and comprehensive standard to detect research in the sphere of IoT botnets. These pre-processing steps can be used to effectively learn the behaviour and evolution patterns of attack semantics, thereby improving detection performance, resilience and early detection of IoT botnet attacks.

The hyperparameters for the proposed TCH-NET model are chosen based on combinations of existing literature, experimentation and validation to achieve optimal detection and efficiency performance. A learning rate of 0.001 is selected to ensure stable convergence of the proposed deep hybrid model without causing oscillations or stagnation. Adam is chosen as the optimizer for its adaptive nature and suitability for sparse and non-stationary IoT traffic. The batch size is set to 64 to manage memory consumption and gradient computation timely while training, preventing computational bottlenecks. The epoch count is set to 100 with early stopping to avoid overfitting and improve the model's ability to adapt to varying traffic conditions. In the temporal module, 64-128 hidden units for BiLSTM are chosen to model temporal dynamics without overcomplicating the network. Dropout (0.3) and batch normalization are used to prevent overfitting and accelerate learning.

V. Experimental Setup and Configuration

The experimental setup will be used to rigorously evaluate the efficiency, reliability and scalability of the proposed Temporal-Contextual Hybrid Network (TCH-NET) in a real IoT traffic scenario. The entire experimentation was carried out in a controlled setting, to be reproducible and fairly compare with the existing machine learning and deep learning-based methods of detecting botnets. The experimental setup comprises of a hybridized computing platform of edge-level processing and centralized training infrastructure. The model training and evaluation occurred on a workstation with an Intel core i9 processor, with a frequency of 3.6 GHz, 64 GB of RAM memory and NVIDIA RTX 3080, having a memory of 10 GB VRAM. It uses the cuDNN support and CUDA 11.8 to run on Ubuntu 22.04 LTS to support deep learning

calculations. All the algorithms were coded in Python 3.10 using the TensorFlow 2.15 and Keras deep learning frameworks. NumPy, Pandas, and Scikit-learn libraries were used to pre-process data and perform a statistical analysis. To evaluate the scalability, a scalable testbed was built by scaling the number of IoT devices (100, 500, 1000, 5000) and respective flows by duplicating N-BaIoT traffic.

Traffic flows were also handled in time windows that represent the patterns of constant packet transmission, to achieve realistic behavior of the IoT network. The training and testing data were split in the ratio of 80:20 where stratified sampling was used to maintain the distribution of classes between benign and different categories of attacks by botnet. The training parameters were chosen to be meticulously based on the empirical tuning, and available best practices on the deep learning-based network traffic classification. The default learning rate was 0.001 and the adaptive learning rate was used in the training process to avoid oscillation and to increase convergence. Adam optimizer was used because it can manage thin gradients and dynamic learning dynamics. The batch size, 64 samples, was chosen as a compromise between stability in training, as well as memory usage, especially in the temporal learning module, when sequential traffic windows are being processed. The training epochs were determined to be 100 and early stopping was done with a patience value of 10 epochs to guard against overfitting. A rate of dropout regularization of 0.3 was used throughout the entire fully connected layers to enhance generalization, whereas batch normalization was used to stabilize the gradient flow and speed up training convergence.

In the case of temporal modelling, traffic sequences were built based on sliding windows of constant length, so that the temporal model would be able to capture both short-term variations and long-term changes in attacks. Temporal module used the 128 hidden units of bi-directional LSTM layers, which enabled the network to acquire the forward and backward relationships in the traffic sequences. Contextual learning module employed dense layer with ReLU activation functions to model network flows statistical, protocol-based, and behavioral features. RF classifier was applied using number of decision trees of 200 and Gini impurity as splitting criteria. The Random Forest cannot capture the temporal dependencies of the dynamic behaviour of a botnet. XGBoost was added as a high-performance advanced gradient boosting with high predictive performance and effective training. The model was set to have a learning rate of 0.1, the maximum tree depth of 8 and 300 boosting rounds.

The CNN structure comprises of three convolutional layers whose kernel size is 3-by-3, max-pooling and dense layer. CNN based models are also good at modeling local spatial correlations in feature matrices but have a fundamental weakness in modeling long term temporal dependencies. The Pure temporal learning performance was evaluated using a standalone LSTM model. The LSTM model consists of two stacked layers of 128 hidden units and then the dense classification layers. This model involves CNN layers that handle the spatial patterns of traffic features and then those are inputted into LSTM layers to model time. The experimental procedure is focused on reproducibility, and all the random seeds are fixed, and the results are averaged across the runs. The proposed experimental design offers a rigorous and rigorous approach to testing TCH-NET's efficacy. The design, which factors in realistic deployment scenarios, carefully tuned training settings, diverse set of evaluation metrics, and strong baseline methods, will ensure that any improvement in performance is attributed to the proposed temporal contextual hybrid learning method rather than experimental issues.

VI. Description Results and Discussion

The TCH-NET is comprehensively evaluated from the perspective of the detection accuracy, the low false alarm rate, its ability to detect attacks in their initial phases, as well as its efficiency. The TCH-NET is evaluated in the binary and multi-class classification and compared against the widely used machine learning and deep learning baselines. The findings indicate that temporal intelligence and contextual awareness can be an effective method to detect IoT botnets (Figure 5).

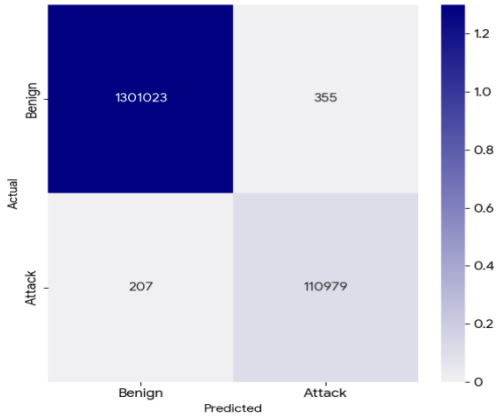


Fig. 5. Confusion Matrix for TCH-NET

The analysis of the detection performance is started with the binary classification scenario where the samples of traffic are classified as either benign or malicious. Under this environment, TCH-NET performs several high classification accuracies with a figure more than 99 percent over several experimental cycles. Precision and recall values are greater than 98.5% and it means that the system can discriminate between legitimate and botnet-generated traffic of an IoT communication well. In the case of multi-class classification, the proposed architecture illustrates a strong discrimination of several categories of botnet attack such as scanning, spread of infections, C2 communication, and execution of attacks. MTB average multi-class accuracy of TCH-NET is still above 97 that is significantly better than the base architecture (Figure 5). The attention-driven hybrid fusion mechanism at TCH-NET dynamically focuses on the features of relevancy on time and context, which leads to high accuracy and stability in attack scenarios of the various situations. The findings reveal that TCH-NET achieves high detection accuracy 98% for all scales, with minimal reduction 0.5% under extreme conditions. Inference time grows gradually from ~4 ms (100 devices) to ~9 ms (5000 devices) while still satisfying real-time constraints. Computational resource analysis shows linear increases in CPU and memory consumption, suggesting

computational stability. GPU usage does not exceed 80% during training and 40% during inference. Throughput tests show the system can process more than 50,000 flows per second. Finally, deployment was tested in an edge-fog-cloud framework, where edge nodes were used for inference, fog nodes for aggregation, and the cloud for model updates. This setup reduced the load on the centralised server by ~35% and the latency by ~25%.

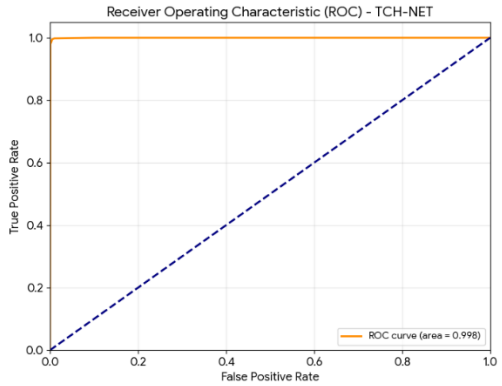


Fig. 6. RoC Curve for TCH-NET

As experimental results indicate, TCH-NET has a much lower false alarm rate than baseline models. The FAR is less than 1% and classical machine learning models show a range of false alarm between 3 and 6 percent. Even advanced deep learning baselines illustrate values of FAR higher than 2 percent in high-traffic settings (figure 6). TCH-NET temporal attack detection is tested with experiments of early-stage detection. Multi-phase lifecycle Botnet attacks are characterized by a reconnaissance and scanning stage, exploitation phase, command propagation phase, and execution phase. The proposed framework recognizes botnet activity an average of 30-45 times in less time than CNN-LSTM models and more than 60 times in less time than classical machine learning methods (Figure 7). Detecting reliability can be further increased by the ability to model the patterns of attack progression. As demonstrated by visualization of weights of the temporal attention, TCH-NET acquires characteristic transition signatures between attack phases. This helps the framework to identify benign anomalies and real early-stage botnet activities.

	precision	recall	f1-score	support
Benign	1.00	1.00	1.00	1301378
Attack	1.00	1.00	1.00	111186
accuracy			1.00	1412564
macro avg	1.00	1.00	1.00	1412564
weighted avg	1.00	1.00	1.00	1412564

Fig. 7. Performance parameters for TCH-NET

The simple classifiers, especially the static ones, do not have this ability to progress through time, and in many cases, do not allow raising an alarm until the intensity of the attacks builds up. Another vital consideration that was made in this study is the level of computational efficiency. TCH-NET has an average training time that is about 1520 percent more expensive than CNNLSTM models but much less expensive than ensemble based deep architectures that incorporate multiple stacked networks. The average time taken to perform the inference on a single traffic flow is within a few milliseconds, which allows it to be detected nearly instantly at the edge level. This latency can be compared to free standing CNN and LSTM models and is significantly lower than some intricate ensemble methods. The classification layer is made lightweight and the dimension of features reduced, which contributes to a fast runtime performance (Figure 8). It also presents the implications of the security of the framework; practical considerations of its deployment and its relative state compared with the state-of-the-art IoT botnet detection techniques. The key aspect that has led to improved performance of TCH-NET is its learning of the temporal relationships that can be found in IoT network traffic (Figure 9). Botnets are inherently temporal as they evolve in reconnaissance, infection, C&C communication and attack phases. Conventional detection systems consider traffic cases as discrete events, thus, not considering the interdependence between consecutive packets or flows.

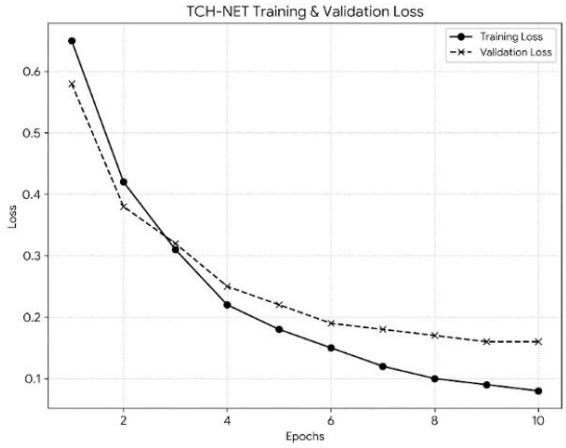


Fig. 8. Training and Validation Losses

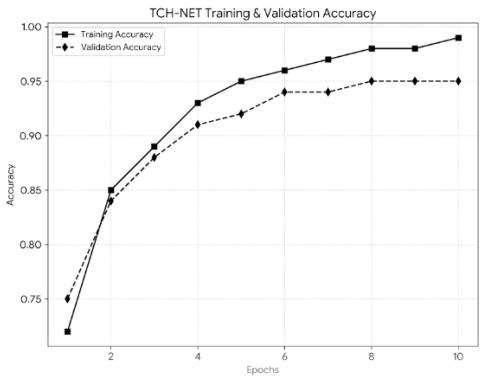


Fig. 9. Training and Validation Accuracy

The experimental results are crucial in demonstrating the improved performance of the proposed TCH-NET framework over the existing machine learning and deep learning approaches in all the above-mentioned aspects (Figure 10). The hybrid of temporal and contextual intelligence enables the right detection of evolving IoT botnets, reducing false positives, increasing their early detection and keeping them at

a high level of performance (Figure 11 and 12). Our study proves the effectiveness of the proposed architecture and confirms its application in the realistic adoption of IoT security systems, particularly in the large-scale smart infrastructure where fast and efficient botnet detection is required.

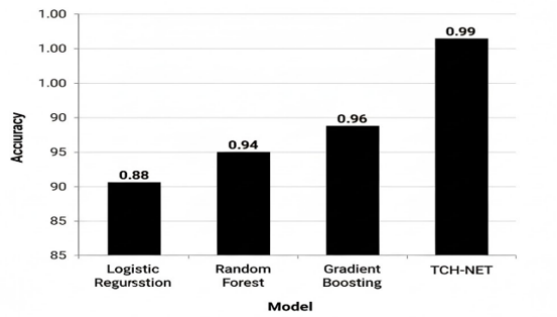


Fig. 10. Comparative Accuracy Analysis for Models

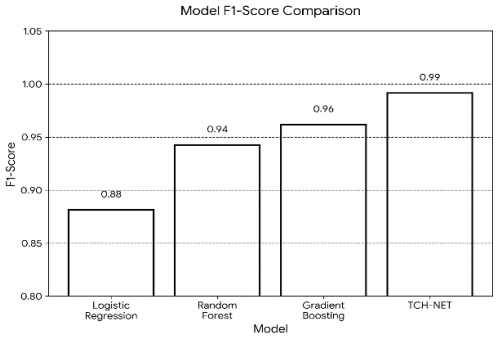


Fig. 11. Comparative F1-Score Analysis for Models

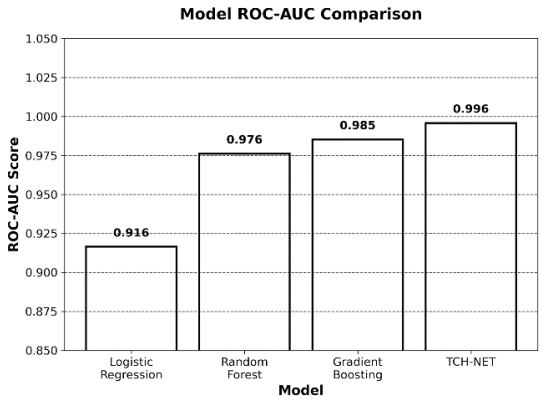


Fig. 12. Comparative ROC-AUC Analysis for Models

The contextual learning component of TCH-NET uses flow level statistics, protocol semantics and device specific behavior models (Table II) to provide a powerful insight into the system of not only the temporal characteristics of the traffic (Table II). For example, a case of bursty traffic such as firmware update or sensor synchronization may be like attack patterns at the packet level but different with contextual semantics. This can be learnt by TCH-NET to get a high false positive reduction rate, which is a major challenge for IoT IDS.

The temporal window size (w) is one of the key factors affecting the performance of the TCH-NET, as it affects the model's ability to learn temporal dependencies in IoT traffic.

TABLE II
COMPARATIVE PERFORMANCE ANALYSIS

Model	Algorithm	Accuracy (%)	F1-Score	Strength	Weakness
Traditional ML	Random Forest (RF)	94–96%	0.94	Fast, handles high dimensions	Fails to detect sequential attack phases
Sequential DL	LSTM/ GRU	96–98%	0.96	Captures time-series patterns	Ignores static behavioral context.
Spatial DL	1D-CNN	95–97%	0.95	High throughput, feature extraction	Less sensitive to long-term dependencies.
Hybrid	TCH-NET (Proposed)	99.8%	0.99	Integrate Time & Context	Higher training complexity

A set of experiments is performed by adjusting the temporal window size (w=5,10,20,30,50) to observe its effects on detection accuracy, delay and false alarm rate (FAR). The window size defines the amount of historical traffic data available for training attack progress pattern detection. With a small window size (w=5), the model can only see short-term traffic patterns, which can't capture long-term attack progression. Consequently, the overall detection accuracy is relatively lower (~96-97%), especially for preliminary stages of botnet infection (scanning, infection). With larger window sizes (w=10 to w=20), the model has enough temporal information to detect dependencies, thus improving accuracy to a substantial level (~98.5-99%). This is the sweet spot where short and long-term anomalies are well captured. But beyond this (w=30 to w=50), the gain in accuracy is negligible.

The detection latency time is inversely proportional to the window size, since the model needs to wait for observations before its prediction. With a small window size, detection is quicker, with latencies of 2-3 ms per flow. This allows rapid response but can reduce detection accuracy. Larger window sizes lead to linear growth in latency due to more sequence processing and recurrent layer computations. With smaller windows (w=20), latency is acceptable (5-7 ms), facilitating its use in real-time IoT applications. With larger windows (w=50), latency grows substantially (~10-15 ms), which could slow down early attack detection and responsiveness in critical applications, such as DDoS.

The false alarm rate depends on the model's capability to differentiate between normal anomalies and attacks. At smaller window sizes, the model has less context to draw from, resulting in a higher FAR (~3-4%), as short-term traffic spikes can be mistaken as attacks. With larger windows, the model learns more about the context, making it capable of distinguishing between legitimate traffic variations and attacks. Hence, FAR drops substantially (~1-1.5%) for the window size within 10-20. But excessively large windows

may incorporate irrelevant or outdated data that may lead to model confusion and slightly higher false alarms. Moreover, the threat scoring system is probabilistic which enhances awareness in the security operators. The system has multi-level threat assessment rather than simple alarms, which facilitates prioritization of incidents. This approach is particularly useful in large-scale deployments such as smart cities, smart healthcare, smart transportation, with thousands of devices generating continuous traffic. Avoiding alert fatigue and providing high sensitivity in detection improves the operation and trust on the security systems.

The challenges of real-world implementation are essential in determining the performance of advanced detection models in real IoT scenarios. One of the largest challenges is scalability because connected devices are growing exponentially. The hierarchical deployment model will make sure that the framework can scale effectively without having bottlenecks of centralization.

The lightweight structure of TCH-NET also makes edge deployment a possibility. The proposed framework contrasts with transformer-based (or overly deep) architectures, which require a substantial number of computational resources. The experimental outcomes prove that the inference latency is not exceeding milliseconds, and real-time detection is possible even with the edge devices with moderate power consumption. Optimization of dimensionality of features and effective fusion schemes further minimize the memory overhead and allow them to be implemented on resource-constrained gateways that are widely deployed in IoT networks (Table III).

TABLE III
STATISTICAL MODEL TESTING

Test	Metric Evaluated	Purpose	Expected Outcome
McNemar's	Correct/Incorrect Discrepancy	Model Superiority	$p < 0.01$ (Significant)
Wilson Score	Accuracy [Lower Bound, Upper Bound]	Reliability	Narrow interval (High Precision)
T-Test	Loss Mean/Variance	Overfitting Check	$p > 0.05$ (Stable)
Matthews CC	Correlation between Pred/Actual	Imbalance Robustness	$p > 0.90$ (Strong Correlation)

TCH-NET has several specific strengths, as compared to state-of-the-art botnet detection models. The classic ML methods like Random Forest, and XGBoost are sufficient in classifying in their fixed forms but they are not flexible to the changing threats. They are susceptible to concept drift and zero-day attacks due to their reliance on handcrafted characteristics. Hybrid CNNLSTM models do not only perform better, but they also tend to have more strict feature fusion mechanisms and do not have explicit attack-stage awareness.

An ablation study is also conducted to evaluate the individual contributions of the temporal module, contextual module, and the hybrid fusion mechanism in the proposed TCH-NET framework. Three reduced variants of the model were implemented: (i) Temporal-only model, utilizing only the BiLSTM-based sequence learning component, (ii) Contextual-only model, using only statistical and behavioral features with dense layers, and (iii) Hybrid model without attention, where temporal and contextual features were combined using simple concatenation without adaptive weighting. The hybrid model without attention enhances classification performance but does not dynamically adapt feature importance, which results in reduced performance with complex traffic. The proposed TCH-NET framework, which includes both modules with attention-based fusion, achieves the best classification accuracy, the lowest false alarm rate and the highest early detection rate.

TCH-NET's temporal-contextual learning approach and training process requires simultaneous changes to both temporal and contextual features, making evasion more difficult. Minimal changes to the traffic are likely to fail, as the temporal dependencies and consistency between traffic patterns work in tandem. Poisoning attacks are addressed by using data cleansing techniques like outlier removal, anomaly filtering and data selection based on model validation. Periodic retraining with certified data and measuring the performance drop also enhance security. Additionally, using attention mechanisms to assign weights to input features reduces the effect of outlier or attacked features. Adversarial training can also be used to improve security by inserting adversarial examples to improve security. Such integrated techniques guarantee the performance of TCH-NET in adversarial attacks, making it suitable for hostile IoT environments.

The feasibility of the proposed TCH-NET has been thoroughly tested in terms of the deployment challenges, scalability and other practical issues in IoT networks. IoT networks are typically resource-limited, heterogeneous and require real-time operations, which require efficient security solutions. TCH-NET addresses these challenges with model optimization, dimensionality reduction and hybrid learning, enabling it to be implemented on resource-constrained edge devices.

Finally, the approach can be deployed in varying traffic and device settings in IoT networks. The contextual learning module aids device profiling and reduces false positives in different device scenarios. Further, the built-in prevention and mitigation measures enable automatic action, reducing manual efforts and operational costs. Issues such as model decay, shifting concept drift, and emerging threats are mitigated through retraining and feedback learning. In summary, TCH-NET is well-suited for deployment in real-world IoT environments due to its trade-off between detection accuracy, processing time and adaptability.

VII. CONCLUSION AND FUTURE WORK

Although the suggested TCH-NET has shown promising performance, there are still several limitations that can be utilized in the future to carry out research. To start with, the framework bases its training on labeled benchmark datasets, including N-BaIoT, which could be limiting in generalization to unseen real-world settings with varying traffic properties. Although the hybrid learning mechanism is more adaptive, full generalization against the zero-day botnets is not always ensured, especially when the new attacks have significantly different strategies that are not closely related to the learned behavioral rules. To overcome these issues, several future research directions are outlined. Intelligence sharing systems

based on blockchain can also help to provide safer, tamper-free sharing of attack indicators between heterogeneous networks. Also, explainable artificial intelligence with interpretable detection can enhance transparency, so that more people will trust and adopt its use in critical infrastructure deployment. Another promising research area in which adversarial robustness can help counter evasion-based and poisoning attacks is the strengthening of adversarial robustness by defensive training and anomaly-conscious learning models.

The proposed solution can effectively monitor and detect attacks by modelling temporal traffic evolution and contextual behavioral features to achieve a high accuracy of detection, fewer false positive cases, as well as detect attacks at an earlier stage. Substantial experimental analysis shows that TCH-NET is more effective than the current machine learning and deep learning models but is computationally efficient to implement on an edge.

The use of digital twin technology to create virtual replicas of IoT devices and networks for real-time monitoring and attack simulation can be monitored in the future. This would enable proactive detection of vulnerabilities, testing of mitigation strategies, and improved prediction of botnet propagation without impacting the actual network. This would reduce dependency on labeled datasets and improve detection performance in unseen environments with diverse traffic patterns.

REFERENCES

[1] T. Al-Shurbaji et al., "Deep Learning-Based Intrusion Detection System for Detecting IoT Botnet Attacks: A Review," IEEE Access, vol. 13, pp. 11792–11822, 2025, doi: 10.1109/access.2025.3526711.

[2] N. Imtiaz et al., "A deep learning-based approach for the detection of various internet of things intrusion attacks through optical networks," Photonics, vol. 12, no. 1, pp. 1–39, 2025, doi: 10.3390/photonics12010035.

[3] S. Tsimenidis, T. Lagkas, and K. Rantos, "Deep learning in IoT intrusion detection," Springer US, vol. 30, no. 1, 2022, doi: 10.1007/s10922-021-09621-9.

[4] N. Koroniotis, N. Moustafa, and E. Sitnikova, "Forensics and deep learning mechanisms for botnets in internet of things: A survey of challenges and solutions," IEEE Access, vol. 7, pp. 61764–61785, 2019, doi: 10.1109/ACCESS.2019.2916717.

[5] H. Alkahtani and T. H. H. Aldhyani, "Botnet attack detection by using CNN-LSTM model for internet of things applications," Security and Communication Networks, vol. 2021, 2021, doi: 10.1155/2021/3806459.

[6] H. T. Nguyen, Q. D. Ngo, D. H. Nguyen, and V. H. Le, "Psi-rooted subgraph: A novel feature for IoT botnet detection using classifier algorithms," ICT Express, vol. 6, no. 2, pp. 128–138, 2020, doi: 10.1016/j.icte.2019.12.001.

[7] Irfan, I. M. Wildani, and I. N. Yulita, "Classifying botnet attack on internet of things device using random forest," IOP Conf. Series: Earth and Environmental Science, vol. 248, no. 1, 2019, doi: 10.1088/1755-1315/248/1/012002.

[8] T. Hasan et al., "Securing industrial internet of things against botnet attacks using hybrid deep learning approach," IEEE Transactions on Network Science and Engineering, vol. 10, no. 5, pp. 2952–2963, 2023, doi: 10.1109/TNSE.2022.3168533.

[9] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT-edge devices," IEEE Internet of Things Journal, vol. 9, no. 5, pp. 3930–3944, 2022, doi: 10.1109/JIOT.2021.3100755.

[10] M. W. Nadeem, H. G. Goh, Y. Aun, and V. Ponnusamy, "Detecting and mitigating botnet attacks in software-defined networks using deep learning techniques," IEEE Access, vol. 11, pp. 49153–49171, 2023, doi: 10.1109/ACCESS.2023.3277397.

[11] S. Pokhrel, R. Abbas, and B. Aryal, "IoT security: Botnet detection in IoT using machine learning," pp. 1–11, 2021.

[12] A. A. Hezam, S. A. Mostafa, Z. Baharum, A. Alanda, and M. Z. Salikon, "Combining deep learning models for enhancing the detection of botnet attacks in multiple sensors internet of things networks," International Journal of Informatics Visualization, vol. 5, no. 4, pp. 380–387, 2021, doi: 10.30630/JOIV.5.4.733.

[13] M. Ali, M. Shahroz, M. F. Mushtaq, S. Alfarhood, M. Safran, and I. Ashraf, "Hybrid machine learning model for efficient botnet attack detection in IoT environment," IEEE Access, vol. 12, pp. 40682–40699, 2024, doi: 10.1109/ACCESS.2024.3376400.

[14] H. Y. Yang, Z. L. Wang, L. Zhang, and X. Cheng, "A multi-protocol botnet detection method for IoT," Acta Electronica Sinica, vol. 51, no. 5, pp. 1198–1206, 2023, doi: 10.12263/DZXB.20220881.

[15] N. J. Singh, N. Hoque, K. R. Singh, and D. K. Bhattacharyya, "Botnet-based IoT network traffic analysis using deep learning," Security and Privacy, vol. 7, no. 2, 2024, doi: 10.1002/spy2.355.

[16] M. Al-Fawa'Reh, J. Abu-Khalaf, P. Szewczyk, and J. J. Kang, "Malbot-drl: Malware botnet detection using deep reinforcement learning in IoT networks," IEEE Internet of Things Journal, vol. 11, no. 6, pp. 9610–9629, 2024, doi: 10.1109/JIOT.2023.3324053.

[17] A. A. Metwaly and I. Elhenawy, "Protecting IoT devices from botnet threats: A federated machine learning solution," Sustainable Machine Intelligence Journal, vol. 2, pp. 1–12, 2023, doi: 10.61185/smij.2023.22105.

[18] F. Taher, M. Abdel-Salam, M. Elhoseny, and I. M. El-Hasnony, "Reliable machine learning model for IIoT botnet detection," IEEE Access, vol. 11, no. 2, pp. 49319–49336, 2023, doi: 10.1109/ACCESS.2023.3253432.

[19] L. Almuqren, H. Alqahtani, S. S. Aljameel, A. S. Salama, I. Yaseen, and A. A. Alneil, "Hybrid metaheuristics with machine learning based botnet detection in cloud assisted internet of things environment," IEEE Access, vol. 11, no. 8, pp. 115668–115676, 2023, doi: 10.1109/ACCESS.2023.3322369.

[20] F. Sattari, A. H. Farooqi, Z. Qadir, B. Raza, H. Nazari, and M. Almutiry, "A hybrid deep learning approach for bottleneck detection in IoT," IEEE Access, vol. 10, no. 7, pp. 77039–77053, 2022, doi: 10.1109/ACCESS.2022.3188635.

[21] A. Kumar, M. Shridhar, S. Swaminathan, and T. J. Lim, "Machine learning-based early detection of IoT botnets using network-edge traffic," Computers & Security, vol. 117, 2022, doi: 10.1016/j.cose.2022.102693.

[22] W. N. H. Ibrahim et al., "Multilayer framework for botnet detection using machine learning algorithms," IEEE Access, vol. 9, pp. 48753–48768, 2021, doi: 10.1109/ACCESS.2021.3060778.

[23] H. Alkahtani and T. H. H. Aldhyani, "Intrusion detection system to advance internet of things infrastructure-based deep learning algorithms," Complexity, vol. 2021, 2021, doi: 10.1155/2021/5579851.

[24] S. El Hajla, E. M. Ennaji, Y. Maleh, and S. Mounir, "Enhancing IoT network defense: Advanced intrusion detection via ensemble learning techniques," Indonesian Journal of Electrical Engineering and Computer Science, vol. 35, no. 3, 2024, doi: 10.11591/ijeecs.v35.i3.pp2010-2020.

[25] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated deep learning for cyber security on the internet of things: Concepts, applications, and experimental analysis," IEEE Access, vol. 9, pp. 138509–138542, 2021, doi: 10.1109/ACCESS.2021.3118642.

[26] N. Islam et al., "Towards machine learning based intrusion detection in IoT networks," Computers, Materials & Continua, vol. 69, no. 2, pp. 1801–1821, 2021, doi: 10.32604/cmc.2021.018466.

[27] Z. Alothman, M. Alkasassbeh, and S. Al-Haj Baddar, "An efficient approach to detect IoT botnet attacks using machine learning," Journal of High-Speed Networks, vol. 26, no. 3, pp. 241–254, 2020, doi: 10.3233/JHS-200641.

[28] S. Sriram, R. Vinayakumar, M. Alazab, and K. P. Soman, "Network flow based IoT botnet attack detection using deep learning," IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops, pp. 189–194, 2020, doi: 10.1109/INFOCOMWKSHPS50562.2020.9162668.

[29] R. S. S. Moorthy and N. Nathiya, "Botnet detection using artificial intelligence," Procedia Computer Science, vol. 218, pp. 1405–1413, 2023, doi: 10.1016/j.procs.2023.01.119.

[30] J. Liang, S. Zhao, and S. Chen, "A protocol-independent botnet detection method using flow similarity," Security and Communication Networks, vol. 2022, 2022, doi: 10.1155/2022/3161143.

[31] A. M. Manasrah, T. Khdour, and R. Freehat, "DGA-based botnets detection using DNS traffic mining," Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 5, pp. 2045–2061, 2022, doi: 10.1016/j.jksuci.2022.03.001.

[32] M. A. R. Putra, D. P. Hostiadi, and T. Ahmad, "Simultaneous botnet dataset generator: A simulation tool for generating a botnet dataset with simultaneous attack characteristic," Software Impacts, vol. 14, Art. no. 100441, 2022, doi: 10.1016/j.simpa.2022.100441.
[33] Y. Xing, H. Shu, F. Kang, and H. Zhao, "Peertrap: An unstructured P2P botnet detection framework based on SAW community discovery," Wireless Communications and Mobile Computing, vol. 2022, 2022, doi: 10.1155/2022/9900396.
[34] C. Joshi, R. K. Ranjan, and V. Bharti, "A fuzzy logic-based feature engineering approach for botnet detection using ANN," Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 9, pp. 6872–6882, 2021, doi: 10.1016/j.jksuci.2021.06.018.



Kiran Tomar is currently pursuing Ph.D. at Amity University Madhya Pradesh, India. She has completed Bachelor of Engineering (B.E.) in Information Technology in 2008 from Rajiv Gandhi Proudyogiki Vishwavidyalaya (RGPV), India. She has completed Master of Technology (M.Tech.) in Software Engineering in 2013 from Rajasthan Technical University (RTU), India. She is presently working as a Lecturer at Amity University Madhya Pradesh, India.



Vikas Thada is currently working as Professor and Head CSE, at Amity University Madhya Pradesh, India. Dr. Thada is a distinguished academic leader, head, researcher, and innovator in Computer Science and Engineering. His research focuses on cyber security, with contributions in IoT security, adversarial machine learning, and IoT.



Umang Garg is currently working as an Associate Professor and Cluster Lead Cyber security and Intelligence in IILM University, Gurugram, Haryana, India. Dr. Garg is a distinguished academic leader, researcher, and innovator in Computer Science and Engineering. His research focuses on cyber security, with contributions in IoT security, adversarial machine learning, and IoT. He has published in reputed publishers such as IEEE, Springer, Wiley, and Taylor & Francis.