

Smart Home Intrusion Detection using a Kepler Optimized Dual Attention Sparse Transformer Network

Thomas Thangam¹

¹Assistant Professor, Department of Process Engineering,
International Maritime College of Oman,
National University of Science and Technology Sohar,
Sultanate of Oman. thomasthangam@gmail.com

ABSTRACT: Smart Home Intrusion Detection Systems (IDS) protect residence by monitoring physical break and network vulnerabilities through integrated sensors and real time alerts. Therefore, in this paper, a Dual Attention Sparse Transformer Network (DASTN) with Kepler Optimization Algorithm (KOA) is proposed for smart home intrusion detection. Data preprocessing in a smart home IDS converts raw, noisy IoT sensor and network traffic logs into a clean format that Deep Learning (DL) models can accurately analyse. A DASTN detects smart home cyber threats by combining spatial temporal feature extraction with reduced computational overhead. The KOA is a physics-based metaheuristic technique used to fine-tune complex model architectures, optimize weights, and select critical hyperparameters. The performance analysis is carried out using python software and assesses the model's performance using key metrics such as precision, recall, F1-score, confusion matrix, and ROC-AUC. The proposed DASTN-KOA uses two classes and the proposed DASTN-KOA achieves high accuracy of 96.67% and better convergence speed. The proposed method outperforms conventional techniques, contributing a robust and reliable tool for smart home intrusion detection system.

Keywords: DASTN, KOA, Smart home IDS, precision, recall, F1-score, confusion matrix, and ROC-AUC.

1. Introduction

An IDS is intended to identify network actions for malicious movements or strategy destructions as safeguarding the digital integrity of smart homes lies the IDS [1]. As a smart home's digital immune system, an intrusion detection system (IDS) continuously looks for irregularities that could indicate possible intrusions. Its main purpose is to detect known and unknown risks, making sure that any illegal or questionable conduct is found and dealt with right away [2]. From basic signature-based indicators that rely on recognized patterns of malicious action to more complex systems that use cutting-edge techniques like anomaly detection and heuristics, traditional intrusion detection systems have evolved [3]. Even with

their improvements, classic intrusion detection systems (IDS) frequently fail to keep up with the complication and originality of contemporary cyber threats, especially in the networked environment of smart homes [4, 5]. Due to their inherent limitations, signature-based IDS are susceptible to zero-day vulnerabilities and unidentified threats because they can only identify assaults using pre-defined signatures [6].

DL is vital for smart home intrusion detection because it automates feature extraction and handles complex, high-volume IoT traffic better than traditional rule-based systems [7]. The conventional techniques such as Convolutional Neural Network-Hyper Parameter Optimization (CNN-HPO) [8], Long Short-Term Memory-

Genetic Algorithm [9] and GhostNet with Bidirectional Gated Recurrent Unit with Artificial Humming Bird Optimization (GhostNet-BiGRU-AHB) have been used for smart home IDS. These techniques have some disadvantages like low convergence speed, less reliability and High response time. To avoid these issues, an DASTN-KOA is proposed for smart home IDS. The contribution of the paper is as follows,

- Data preprocessing is developed to transform raw, noisy, and intrusion network traffic into a clean, standardized, and manageable format that optimizes DL model accuracy and real-time detection performance.
- Data visualization is used to transform complex numerical metrics and categorical model predictions into intuitive visual formats.
- DASTN is proposed for smart home IDS to accurately predict and categorize attacks by automatically extracting complex data patterns.
- KOA is proposed to optimize complex model hyperparameters and structural configurations by simulating planetary motion.

2. Recent Works

Intrusion detection using CNN-HPO networks have been presented in [8] to analyse intrusion detection. hyper-parameter optimization (HPO) based on multi-objective discrete extremal optimization (MODEO) to automatically design a lightweight CNN for the intrusion detection task of IIoT, abbreviated as MODEO-CNN. A novel hybrid variable-length encoding strategy is developed by combining binary and integer encoding to characterize both the neural architectures including the number of blocks, the blocks-based network topology and the corresponding architecture parameters in CNN block, and some important hyper-parameters

including batch size, learning rate, weight optimizer and regularization. [9] proposes LSTM-GA, an adaptive IoT IDS for smart network management using NetFlow and IIoT data. This framework leverages pre-trained CNN models to convert data into images, enabling effective classification. GA optimizes hyperparameters to improve model flexibility and performance, while a soft voting ensemble ensures robust aggregation of predictions. The hybrid GhostNet-BiGRU-AHO has been proposed in [10] for smart intrusion detection. The local model updates are encoded using homomorphic encryption, which advances statistics safety and confidentiality. To diminish communication sequences throughout statistics accumulation, server-side update accumulation and cooperative optimization are employed. However, it produces high computational costs, severe training instability, data sparsity issues, and vulnerability to adversarial attacks, making real-time deployment difficult and model optimization complex. To overcome these drawbacks, an DASTN-KOA is proposed in this paper for smart home intrusion detection.

3. Proposed Work Explanation

The proposed block diagram of DASTN-KOA for smart home intrusion detection system is represented in figure 1. The process systematically transforms raw data into optimized predictive outputs through five core stages they are data preprocessing & visualization. The workflow begins with raw data entering the data preprocessing stage, which focuses on structural preparation and data imputation to handle missing values. The output from the data preprocessing is fed into the data visualization process, which allows for initial exploratory data analysis to understand underlying patterns and distributions. In feature engineering, the processed data undergoes normalization to scale numerical ranges consistently, feature selection to isolate the most informative attributes and reduce dimensionality.

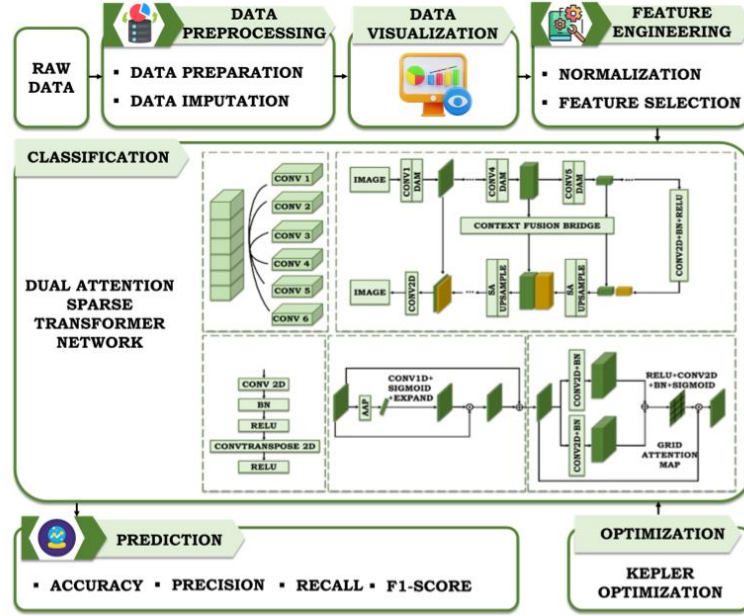


Figure 1: Proposed block diagram of DASTN-KOA for smart home intrusion detection

DASTN is the core architectural component of the framework. It utilizes a multi-branch DL network that processes input data using a series of convolutional layers. The architecture features a specialized context fusion to combine features across different layers, incorporates spatial or channel-based attention through attention blocks, and employs a grid attention map module to dynamically weight critical feature regions. KOA is a metaheuristic approach, which fine-tunes the hyperparameters or structural weights of the classification model to maximize its learning efficiency. In the final phase, the trained network generates a prediction. The model's performance is rigorously measured against standard classification metrics, specifically accuracy, precision, recall, and F1-Score.

3.1 Modelling of Data Visualization

The process of turning raw numbers into clear pictures like charts and graphs is named as data visualization. To calculate key values, map coordinates, and show trends to understand information quickly, data visualization uses mathematical formulas.

Mathematical expression for mean (average) to check data center:

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i \quad (1)$$

Mathematical expression for min-max feature scaling:

$$x_{normalized} = \frac{x_i - \min(X_{train})}{\max(X_{train}) - \min(X_{train})} \quad (2)$$

Crucially, the min and max values are calculated only from the training data (X_{train}). The same parameters are then used to transform the test data.

Choose variables for the X and Y axes and convert data points to pixel coordinates on a screen. Linear scaling equation for pixel position is derived as:

$$P = \frac{x - x_{Min}}{x_{Max} - x_{Min}} \times pixelwidth \quad (3)$$

Instead of scanning through thousands of rows in a spreadsheet, visualization tools convert data into intuitive shapes and colors, making systemic flaws immediately apparent. An DASTN is used after initial visualization or feature extraction to dynamically capture global context and complex long-range dependencies that standard convolutional filters miss. It refines visual or spatial data by adjusting to varying shapes, noise, or multi-scale structures.

3.2 Mathematical modelling of DASTN

A DASTN is an advanced neural network architecture designed to solve the heavy quadratic

computational complexity. It achieves accuracy by combining two distinct attention paradigms

such as dual attention and sparse attention. Architecture of DASTN is represented in figure 2.

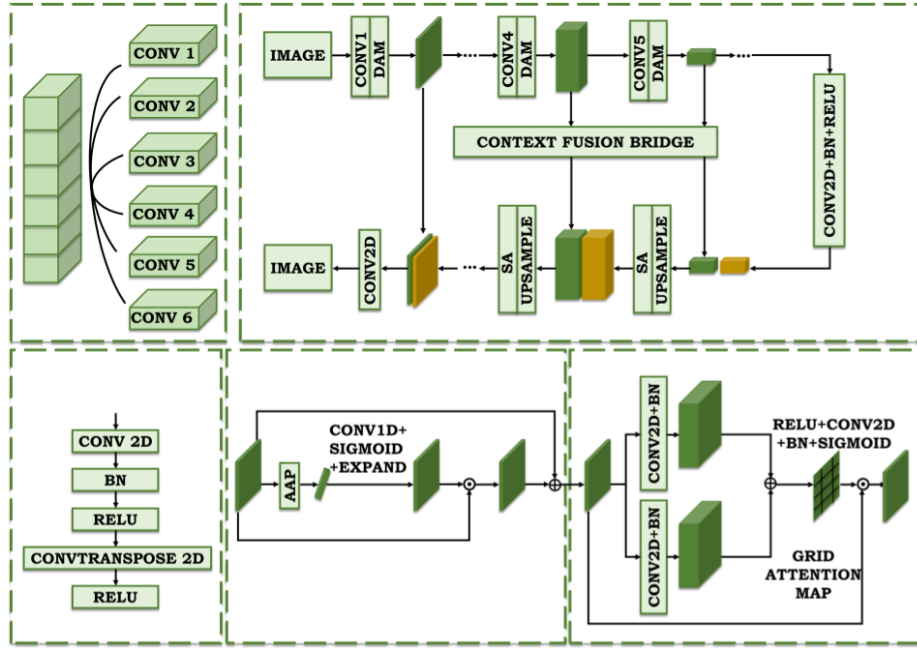


Figure 2: Architecture of DASTN

3.2.1 Input Projection and Feature Embedding:

An input sequence or feature map X in $X \in R^{N \times D}$. It is first projected into query, key and value matrices via learned weight matrices $W_Q, W_K, W_V \in R^{D \times D}$

$$Q = XW_Q, K = XW_K, V = XW_V \quad (4)$$

3.2.2 Applying the Sparse Masking Matrix

An attention scores are computed for every pair of tokens. A sparse transformer introduces a predefined or adaptive sparsity index pattern, defining a restricted subset of indices. Tokens only attend to immediate neighbours within a specific block size

Mathematically, this generates a binary masking matrix $M \in R^{N \times N}$.

$$M_{ij} = \begin{cases} 0, & \text{if } j \in A_i \\ -\infty, & \text{otherwise} \end{cases} \quad (5)$$

3.2.3 Dual-Branch Attention Computation

The network connects the masked representations through two parallel or sequential paths to capture multidimensional interdependencies.

Sparse Spatial/Temporal Attention branch captures long-range context across the length of the sequence or the image grid. The sparse scaled dot-product attention equation is defined as:

$$\text{Sparse} - \text{Attn}_{\text{spatial}}(Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{D_k}} + M \right) V \quad (6)$$

Simultaneously, the channel/spectral attention evaluates connections along the latent dimension D_k rather than the sequence dimension (N) to filter noise and preserve high-frequency features. Instead of computing similarity across rows, it evaluates similarity across columns.

$$\text{Attn}_{\text{channel}}(Q, K, V) = Q \cdot \text{Softmax} \left(\frac{K^T V}{\sqrt{N}} \right) \quad (7)$$

3.2.4 Gated Dual Feature Fusion

The features derived from both spatial and channel contexts must be safely integrated. A dynamic gating module controls how much weight to allocate to each branch.

$$F_{\text{Spatial}} = \text{Sparse} - \text{Attn}_{\text{spatial}}(Q, K, V) \quad (8)$$

$$F_{\text{Channel}} = \text{Attn}_{\text{channel}}(Q, K, V) \quad (9)$$

The outputs are fused using learnable scale coefficients (α and β).

$$Y_{Fused} = \alpha \odot F_{spatial} + \beta \odot F_{channel} \quad (10)$$

3.2.5 Post-Attention Block Processing

To complete the Transformer block, the network applies layer normalization and passes the output through a localized Feed-Forward Network (FFN) utilizing skip connections:

$$H = \text{LayerNorm}(Y_{Fused} + X) \quad (11)$$

$$O = \text{LayerNorm}(\text{FFN}(H) + H) \quad (12)$$

A DASTN provides real-time, low-latency, and highly accurate threat identification by efficiently isolating malicious signals from redundant smart home IoT traffic.

Table 1: Performance metrics of DASTN

Performance metrics	values
Accuracy	0.9667
Precision	0.9667
Recall	0.9667
F1-Score	0.9667
ROC-AUC	0.9949
Cohen's kappa	0.9333
MCC	0.9333
Log loss	0.0905

Table 1 shows the performance metrics of DASTN. The accuracy of proposed DASTN is 96.67%. Optimization is needed after an initial classification model, which is used to fine-tune parameters, reduce errors, and maximize predictive accuracy on unseen data.

3.3 Modelling of KOA

The KOA is a meta-heuristic optimizer inspired by Kepler's laws of planetary motion with a DASTN provides an automated pipeline for tuning hyperparameters. In this system, KOA acts as a global black-box optimizer exploring a complex search space, while the Dual Attention Sparse Transformer represents the objective evaluation function. Flow chart of KOA for smart home intrusion detection is represented in figure 3.

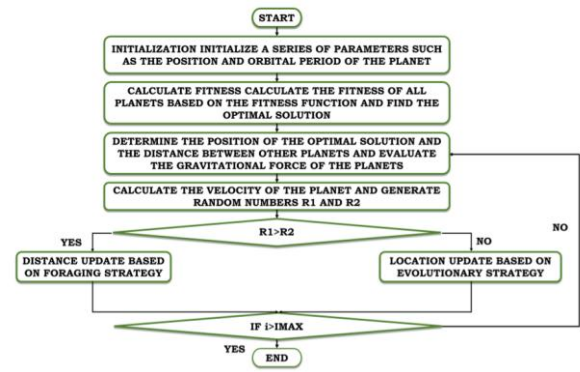


Figure 3: Flowchart of KOA for smart home intrusion detection

3.3.1 Define the Hyperparameter Space and Objective Function

First, map the hyperparameters of the Dual Attention Sparse Transformer to a search vector. Let D be the dimension of the search space.

$$x = [\eta, \lambda, p_{drop}, k_{sparse}, \alpha_{dual}] \quad (13)$$

Where η is the learning rate, λ is the weight decay, p_{drop} is the dropout rate, k_{sparse} is the sparsity constraint parameter (e.g., top-k keys or α -entmax boundary), and α_{dual} governs the blending ratio between the dual attention pathways.

The objective function is defined to minimize validation loss or maximize validation accuracy/F1-score.

$$f(x) = \text{EvaluateTransformer}(x) \quad (14)$$

3.3.2 Initialize the Population

Generate an initial population of N candidate hyperparameter configurations randomly or using a low-discrepancy sequence (e.g., Good Point Set).

$$x_i = x_{Min} + \text{rand}(0,1) \cdot (x_{Max} - x_{Min}) \quad (15)$$

Evaluate each planet's fitness x_i by training the network for a short proxy epoch or using multi-fidelity evaluation.

3.3.3 Compute Physical Parameters

In KOA, the mass M_i of each planet (configuration) is determined by its performance:

$$M_i = \frac{f(x_i) - \text{worst}}{\sum_{j=1}^N (f(x_j) - \text{worst})} \quad (16)$$

Calculate the Euclidean distance R_i between planet i and the Sun x_{sun} :

$$R_i = \|x_{sun} - x_i\|_2 \quad (17)$$

Compute the universal gravitational attraction force $F_i(t)$ exerted by the Sun on planet i at iteration t :

$$F_i(t) = G(t) \cdot \frac{M_{sun} \cdot M_i}{R_i^2 + \epsilon} \quad (18)$$

Where ϵ is a small constant to prevent division by zero, and $G(t)$ is the gravitation constant decreasing over time to transition from exploration to exploitation:

$$G(t) = G_0 \cdot e^{-\lambda \frac{t}{T_{Max}}} \quad (19)$$

3.3.4 Update Velocity and Position

According to Kepler's second law, planets accelerate as they get closer to the Sun (perihelion) and update the orbital velocity vector v_i for the hyperparameter settings:

$$v_i(t+1) = v_i(t) + F_i(t) \cdot (x_{sun} \cdot x_i(t)) \quad (20)$$

Position transformation calculates the orbital eccentricity factor e_i and use it to update the position (hyperparameters):

$$x_i(t+1) = x_i(t) + U \cdot v_i(t+1) \cdot \cos(\theta_i) + V \cdot (x_{sun} - x_i(t)) \cdot \sin(\theta_i) \quad (21)$$

Where U and V are stochastic coefficients, and θ_i represents the polar orbital angle updated at each step to simulate an elliptical trajectory through the hyperparameter space.

3.3.5 Apply Boundary Constraints

Ensure that the updated hyperparameters fall within physically and computationally viable limits.

$$x_i(t+1) = \max(x_{Min}, \min(x_{Max}, x_i(t+1))) \quad (22)$$

3.3.6 Update the Best Solution

Evaluate the new configurations. If any planet $x_i(t+1)$ achieves a lower validation error than the current Sun x_{sun} , update the Sun's position:

$$x_{sun} = x_i(t+1) \quad (23)$$

3.3.7 Iteration and Termination

Repeat Steps 3 through 6 until the maximum number of generations T_{Max} is reached, or the validation accuracy converges. Extract x_{sun} as the globally optimal hyperparameter configuration for the final training run of a DASTN.

4. Results and Discussion

The DASTN-KO is proposed for smart home intrusion detection, which is evaluated using python software. The name of dataset used for simulation is smart home intrusion detection dataset and includes two classes, they are yes and no.

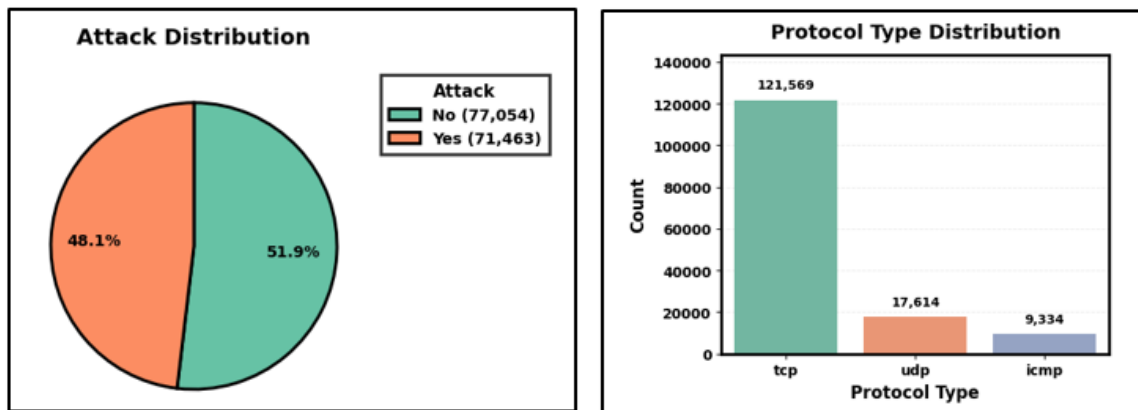


Figure 4: Attack distribution and protocol type distribution

Figure 5 shows the attack distribution and protocol type distribution. It shows that 51.9% of the traffic consists of normal activity, which equates to a count of 77,054 instances. The remaining 48.1% represents active cyberattacks, accounting for 71,463 instances, indicating a relatively balanced dataset between normal and malicious traffic. The chart on the right, titled Protocol Type Distribution, is a bar graph that breaks down the total network traffic by transmission protocols. It measures the volume of each protocol type on the vertical axis against the protocol types on the horizontal axis. TCP is overwhelmingly the most dominant protocol, registering a high count of 121,569. In comparison, UDP sees substantially lower usage with 17,614 instances, while ICMP is the least utilized protocol in this dataset, recording just 9,334 instances.

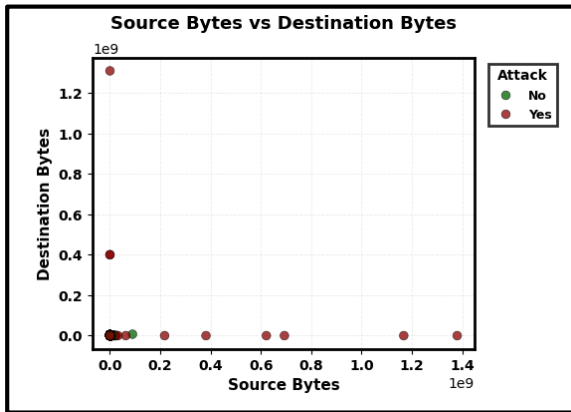


Figure 5: Source bytes vs destination bytes

Figure 6 shows the diagram between source and destination bytes. The scatter plot shows Source Bytes vs Destination Bytes to distinguish network attacks from normal traffic. Both axes use scientific notation scaled to 10^9 bytes (1 GB). Normal traffic stays near the origin, while attacks exhibit two distinct, extreme anomalies. One set of attacks spikes vertically near 0 Source Bytes but peaks at 1.3×10^9 destination bytes. Another set stretches horizontally near 0 Destination Bytes, reaching up to 1.4×10^9 source bytes.

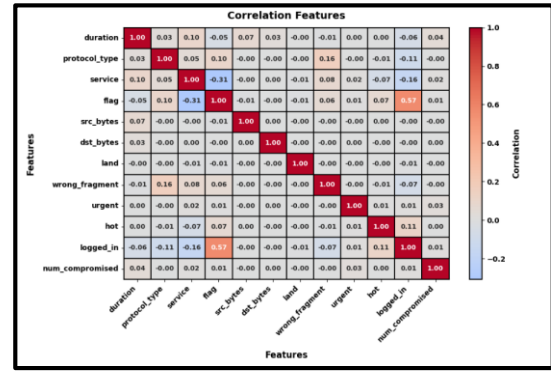


Figure 6: Correlation heatmap

Figure 7 shows a correlation matrix heatmap displaying the linear relationships between twelve distinct features, where values range from -0.21 to 1.00. The diagonal elements show a perfect positive correlation of 1.00, as each feature is perfectly correlated with itself. Most feature pairs exhibit weak relationships near 0.00, indicating little to no linear dependence between them. However, a noticeable positive correlation of 0.57 exists between flag and logged-in, which is highlighted by a light orange shade on the colour scale. In contrast, the strongest negative relationship is found between service and flag at -0.21.

Table 2: Classification report

Classes	Precision	Recall	F1-Score	support
No	0.9678	0.9681	0.9679	15411
Yes	0.9656	0.9652	0.9654	14293
Accuracy			0.9667	29704
Macro avg	0.9667	0.9667	0.8894	29704
Weighted avg	0.9667	0.9667	0.8894	29704

Table 2 shows the classification report, which includes two classes, they are no and yes. The precision of class no is 0.9678, and yes is 0.9656. The recall of class no is 0.9681 and yes is 0.9652. The F1-Score of class no is 0.9679 and yes is 0.9654. The classification report shows the accuracy of proposed DASTN-KO is 96.67%.

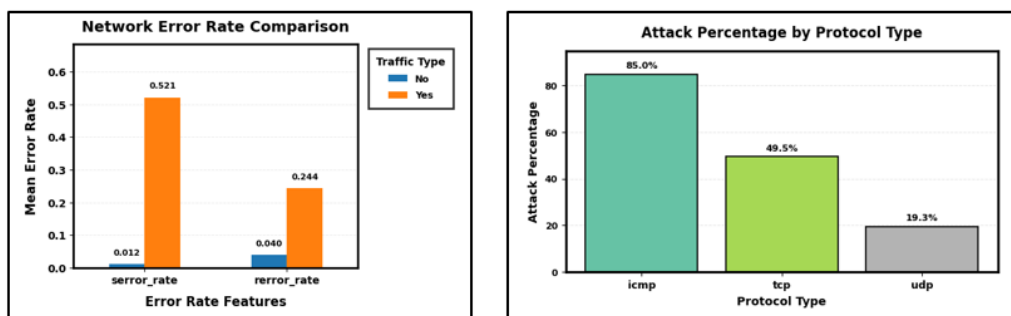


Figure 7: Network error rate comparison and attack percentage by protocol type

Network error rate comparison and attack percentage by protocol type are shown in figure 8. For the `error_rate` feature, the mean error rate is extremely low for the “No” traffic type at 0.012, while it spikes drastically to 0.521 for the “Yes” traffic type. For the `error_rate` feature, the “No” traffic type shows a mean error rate of 0.040,

whereas the “Yes” traffic type increases significantly to 0.244. The `icmp` protocol experiences the highest percentage of attack traffic at 85.0%. This is followed by the `tcp` protocol, which registers an attack percentage of 49.5%. The `udp` protocol accounts for the lowest share of attack traffic among the three, standing at 19.7%

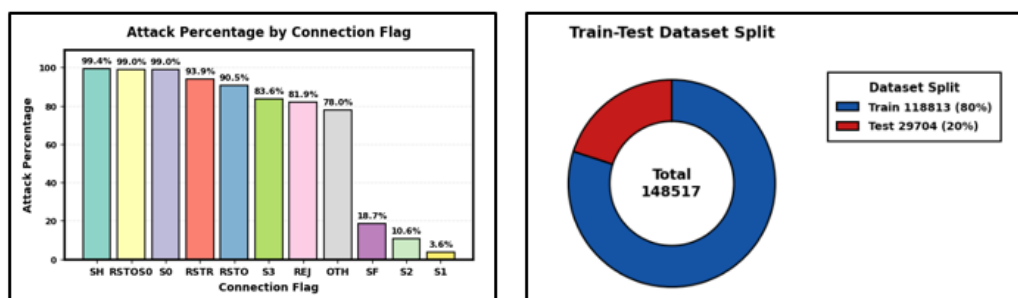


Figure 8: Attack percentage by connection flag and train-test dataset split

Figure 9 shows the Attack percentage by connection flag and train-test dataset split. High-risk flags like SH (99.4%), RSTOS0 (99.0%), and S0 (99.0%) consist almost entirely of malicious traffic, followed closely by RSTR (93.9%), RSTO (90.5%), S3 (83.6%), REJ (81.9%), and OTH (78.0%). In contrast, connection flags like SF (18.7%), S2 (10.6%), and S1 (3.6%) exhibit

significantly lower attack percentages, indicating they mostly represent normal network traffic. Train-test split shows a total of 148,517 instances, which are divided into a standard 80/20 ratio. Specifically, 118,813 samples (80%) are allocated to the Train dataset to train the predictive model, while the remaining 29,704 samples (20%) are reserved for the Test dataset to evaluate its performance.

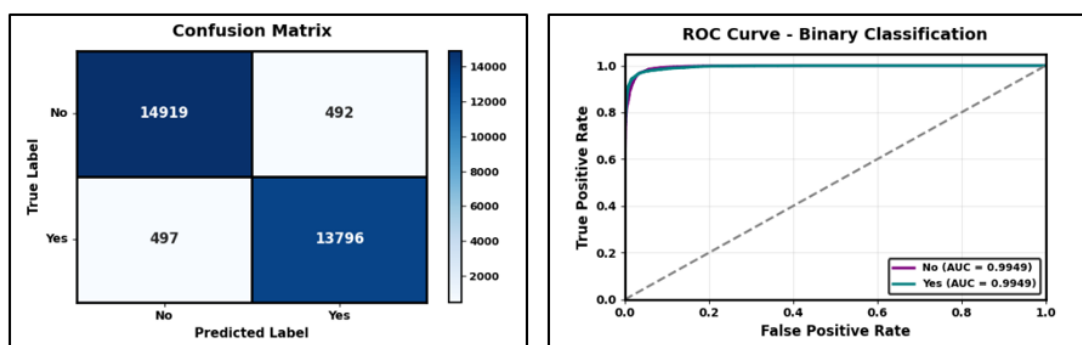


Figure 9: Confusion matrix and ROC Curve

Figure 10 shows the confusion matrix and ROC curve. The confusion matrix shows that the model demonstrates high predictive accuracy, successfully identifying 14,919 true negatives and 13,796 true positives, while making very few errors, with only 492 false positives and 497 false negatives. This strong performance is further

supported by the ROC Curve, which plots the True Positive Rate against the False Positive Rate; its steep rise toward the top-left corner and an exceptionally high AUC of 0.9949 for both classes indicate that the model possesses near-perfect discriminative ability to distinguish between the "Yes" and "No" classes.

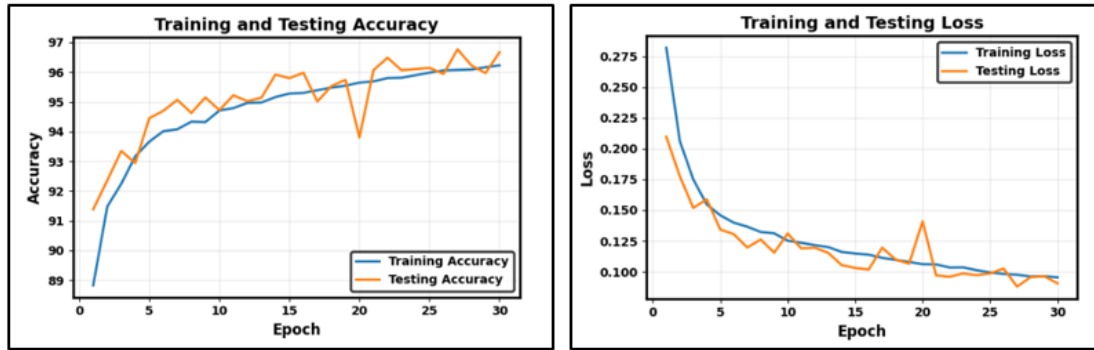


Figure 10: Accuracy and loss of proposed DASTN-KO

The accuracy and loss of proposed DASTN-KO are represented in figure 11. The accuracy of proposed DASTN-KO is 96.67% and the loss of proposed DASTN-KO is 10%, which demonstrates the effectiveness of proposed system.

Table 3: Comparison of response time

Method	Response time
CNN-HPO [8]	0.7s
LSTM-GA [9]	0.6s
GhostNet-BiGRU-AHO [10]	0.5s
Proposed DASTN-KOA	0.1s

Table 2 shows the comparison of response time. The response time of proposed DASTN-KOA is 0.1s. Lower response time (faster inference latency) in an DASTN-KOA classification model enables real-time edge deployment, reduced compute costs, and instant decision-making.

Table 4: Performance comparison

Methods	Precision	Recall	Accuracy
CNN-HPO [8]	94.32%	92.45%	94%
LSTM-GA [9]	94.78%	93%	94.67%
GhostNet-BiGRU-AHB [10]	95.34%	95%	95.34%
Proposed DASTN-KO	96.67%	96.67%	96.67%

Table 4 shows the performance comparison of the accuracy of proposed DASTN-KO is 88%. High accuracy in an DASTN-KO for smart home intrusion detection ensures precise predictions of attacks. The suggested methodology offers a reliable and safe way to detect intrusions, protecting sensitive data while increasing detection precision.

5. Conclusion

In this paper, a hybrid DASTN-KO is proposed for smart home intrusion detection. The performance analysis is carried out using python software. DASTN-KO are effective at classifying intrusion detection, achieving 96.67% accuracy. Utilizing optimized feature sets allows for efficient, real-time detection without requiring massive, computationally expensive models. The developed method is compared with existing methods CNN-HOP, LSTM-GA and GhostNet-BiGRU-AHO in terms of precision, recall and F1-Score. The DL models, DASTN-KO offer high accuracy of 96.67%, and less response time of 0.1s in smart home intrusion detection. The future scope of smart home intrusion detection focuses on localized artificial intelligence, edge

computing, and multi-sensor data fusion to maximize privacy and eliminate false alarms.

References

1. Kang-Di Lu; Yao-Wei Yang; Guo-Qiang Zeng; Chen Peng; Guang-Gang Geng; Jian Weng, Year: 2025, "BPSO-AHDL-IDS: Binary Particle Swarm Optimization-Based Automated Hybrid Deep Learning Model for Intrusion Detection of Internet of Things", IEEE Transactions on Automation Science and Engineering, Vol: 22.
2. Fisayo Sangoleye; Jay Johnson; Eirini Eleni Tsiropoulou, Year: 2024, "Intrusion Detection in Industrial Control Systems Based on Deep Reinforcement Learning", IEEE Access, Vol: 12.
3. Mohamad Khayat; Ezedin Barka; Mohamed Adel Serhani; Farag Sallabi; Khaled Shuaib; Heba M. Khater, Year: 2025, "Reinforcement Learning With Deep Features: A Dynamic Approach for Intrusion Detection in IoT Networks", IEEE Access, Vol: 13.
4. Goce Stevanoski; Aleksandar Risteski; Nevena Serafimova; Marko Porjazovski; Mitko Bogdanoski, Year: 2026, "A Two-Stage Pareto-Driven Framework for Adaptive Resource Optimization in Lightweight Intrusion Detection Systems", IEEE Access, Vol: 14.
5. Sandeepkumar Racherla; Prathyusha Sripathi; Nuruzzaman Faruqui; Md Alamgir Kabir; Md Whaiduzzaman; Syed Aziz Shah, Year: 2024, "Deep-IDS: A Real-Time Intrusion Detector for IoT Nodes Using Deep Learning", IEEE Access, Vol: 12.
6. Yue Li; Jiale Zhang; Yiting Yan; Yutian Lei; Chang Yin, Year: 2024, "Enhancing Network Intrusion Detection Through the Application of the Dung Beetle Optimized Fusion Model", IEEE Access, Vol. 12.
7. Manohar Srinivasan; Narayanan Chidambaram Senthilkumar, Year: 2025, "Intrusion Detection and Prevention System (IDPS) Model for IIoT Environments Using Hybridized Framework", IEEE Access, Vol: 13.
8. Kang-Di Lu; Jia-Cheng Huang; Guo-Qiang Zeng; Min-Rong Chen; Guang-Gang Geng; Jian Weng, Year: 2025, "Multi-Objective Discrete Extremal Optimization of Variable-Length Blocks-Based CNN by Joint NAS and HPO for Intrusion Detection in IIoT", IEEE Transactions on Dependable and Secure Computing, Vol: 22, No: 4.
9. Jing Li; Mohd Shahizan Othman; Xugang Ying; Dina S. M. Hassan; Hewan Chen; Lizawati Mi Yusuf, Year: 2026, "Adaptive NetFlow IIoT Intrusion Detection With Deep Transfer Learning, Genetic Optimization, and Ensemble Methods for Network Management", IEEE Transactions on Network and Service Management, Vol: 23.
10. Om Kumar Chandra Umakantham; Sudhakaran Gajendran; Suguna Marappan, Year: 2024, "Enhancing Intrusion Detection Through Federated Learning With Enhanced Ghost_BiNet and Homomorphic Encryption", IEEE Access, Vol: 12.