International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A NOVEL ALGORITHMIC FRAMEWORK FOR SECURING IOT TRUST MANAGEMENT AGAINST EMERGING TRUST-RELATED ATTACKS

[*]Satish Kamble[1], Dr. Surendra Mahajan[2], Kimi Ramteke[3], Bhavana Chandramani Julme[4]

[1]Research Scholar, Department of Computer Engineering, SKNCOE, SPPU, Pune, India
[2]Department of Information Technology, PVG's College of Engineering, Technology and Management, SPPU, Pune, India
[3]Department of Computer Engineering, International Institute of Information Technology, Hinjewadi, SPPU, Pune, India
[4]Department of Computer Engineering, PVG's College of Engineering, Technology and Management, SPPU, Pune, India

[1*]satkam53@gmail.com, [2]samahajan@yahoo.com , [3]kimiramteke16@gmail.com, [4]bhavanachan@gmail.com

Abstract

The Internet of Things (IoT) is spreading very quickly and has connected and automated a huge range of areas, from transportation and healthcare to industrial systems. IoT networks, on the other hand, are open to a variety of trust-based threats that can slow down systems, stop people from talking to each other, and make security less reliable because they are spread out and different. Attacks like blackhole, grayhole, floods, and TDMA-based outages change trust measures and packet-forwarding behaviours, which has a direct effect on trust management systems. To make IoT trust management safer from these new dangers, this study suggests a new algorithmic framework that combines advanced feature selection and mixed classification methods to improve the accuracy of detection. The first step in the method is to use the WSN-DS dataset, which has examples of attacks on wireless sensor networks that are related to trust and attacks that are not related to trust. As part of data preparation, columns are dropped, labels are encoded, and one-hot encoding is done. Next, Min-Max normalization is used to make sure that the size of all the features is the same. Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) are used alone and together to reduce the number of dimensions while keeping their prediction power. Trust3Net is a mixed PSO+GA method that improves model generalization by choosing the best feature groups. Several classification methods are looked at in the study. These are Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Neural Networks, Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN). Two suggested deep learning-based designs are created: Trust4Net and Trust3Net. Trust3Net uses a dual feature selection approach to improve its performance. Comparative research shows that combining metaheuristic feature selection with deep learning makes IoT trust management systems much more resistant to new threats. The suggested framework provides a strong defence system that can be expanded and changed to fit real-

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

world IoT operations where maintaining trust is important for maintaining operating efficiency and security.

Keywords: IoT Trust Management, Trust-Related Attacks, Feature Selection, Particle Swarm Optimization, Genetic Algorithm, Deep Learning Models

I. Introduction

The Internet of Things (IoT) has quickly become a fundamental shift in the way technology works. It makes it possible for billions of devices, sensors, and apps to join seamlessly across many fields, including smart cities, healthcare, agriculture, transportation, and industrial automation. IoT systems have changed how things work and how people experience them by letting data be collected, processed, and smart decisions made in real time. However, the widespread use of IoT in vital systems has also increased the attack area, leaving these networks open to a wide range of cyber dangers. In contrast to standard computer environments, IoT devices often have limited resources, work in spread out and different environments, and don't come with strong security features by default. This makes it very hard to ensure trust, reliability, and security. Trust management is one of the most important problems in IoT environments. This means figuring out how to make sure that devices can communicate reliably with each other even though their security settings and habits are different [1]. Trust management systems are very important for lowering risks because they constantly check how devices are acting, find bad nodes, and make sure that exchanges are safe. But because IoT settings are so complicated and always changing, trust management can be attacked. In these attacks, bad guys change trust measures on purpose to make networks less reliable or to stop people from talking to each other. These kinds of threats not only make route and resource distribution less effective, but they also make the network less secure generally [2].

Trust-related attacks in IoT networks include many bad actions, such as blackhole attacks, in which hacked nodes falsely report the best routes before dropping data packets; flooding attacks, in which too much traffic overwhelms the network; grayhole attacks, in which some packets are dropped to avoid being caught; and TDMA-based timing manipulation attacks, which mess up the allocation of communication slots in trust-aware MAC protocols [3]. These risks go after the core logic of trust computation, which can lead to bad route choices, lower quality of service, and more secondary attacks being possible. To find these kinds of threats, you need smart, flexible, and computationally efficient methods that can work in a variety of IoT situations [4, 5]. Most of the time, set limits, rule-based tracking, or simple statistical models are used in traditional trust management methods. These methods work well against known attack patterns, but they aren't very good at adapting to new dangers and zero-day attack tactics. Also, they might not take into account the complicated, high-dimensional nature of IoT traffic data properly, which makes them hard to scale and find problems [6]. Machine learning (ML) and deep learning (DL) methods have become very useful for improving IoT security recently. This is because they can learn non-linear trends, change with new attack

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

behaviours, and handle big, diverse datasets. Their success, on the other hand, depends a lot on the quality and usefulness of the input features [7, 8].

Large datasets with unnecessary or duplicate traits not only add to the work that needs to be done, but they also run the risk of overfitting, which makes it harder for the recognition model to work in other situations. So, choosing the right features has become an important part of making trust-related attack detection systems that work well and correctly. When it comes to advanced methods, metaheuristic optimisation algorithms like Particle Swarm Optimisation (PSO) and Genetic Algorithms (GA) have become popular because they can quickly find the best or almost best feature groups in large search spaces [9]. PSO uses the way groups act together to find the best answers over and over again, while GA uses evolutionary ideas like selection, crossing, and variation to look for new combos of features. When used together or separately, these methods can greatly improve the accuracy of recognition while making models simpler [10]. Based on these findings, this study suggests a new way to protect IoT trust management systems from new trust-related threats using an algorithmic approach that combines PSO and GA-based feature selection with the most up-to-date classification models. The suggested framework is tested on the WSN-DS dataset, which is open to the public and has labelled examples of several types of IoT attacks [11]. It starts with organised preprocessing, normalisation, and feature optimisation. Next, it uses a number of classification methods, such as Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Neural Networks, Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN). There are two new models introduced: Trust4Net uses deep learning designs to find trust attacks, and Trust3Net uses a mixed PSO+GA feature selection approach to make it work better.

II. Literature Review

1. Overview of IoT Trust Management

When it comes to the Internet of Things (IoT), trust management is what checks, maintains, and updates how trustworthy devices, nodes, and communication lines are in a network. Because IoT settings are diverse, spread out, and always changing, trust management has become an important part of making sure that devices can operate safely and reliably with each other [12]. Trust management is different from traditional security methods like encryption and identification because it focusses on evaluating behaviour by keeping an eye on things like reaction times, data accuracy, packet transfer rates, and how often people consistently participate in network chores. There are two ways to get trust: directly and indirectly. Directly, a node can judge its neighbours based on how they interact with it, and indirectly, it can get trust numbers from what other trusted nodes say. These sources are often used by modern trust management systems to create combined trust scores. This lets them make smart choices about routes, access control, and working together. IoT networks with limited resources make trust management even more important because centralised tracking isn't possible and bad nodes can easily pretend to be genuine users [13, 14]. There are different types of trust models used

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

in different areas of technology. These include probabilistic and Bayesian methods, as well as fuzzy logic-based systems. Each type is better at dealing with error, missing data, or changing behaviours. A well-thought-out trust management system not only makes the network more resilient, but it also stops false information from spreading and limits the power of nodes that have been hacked [15, 16, 17]. IoT trust management does face some big problems, though. These include making sure that trust calculations can be done on very large networks, that they can handle collusion and recommendation-based trickery, and that they can change to new or changing attack strategies.

2. Classification Of Trust-Related Attacks in IoT

IoT trust-related hacks take advantage of flaws in the way trust is calculated and maintained, which lets bad actors change trust scores, stop contact, or avoid being caught. These attacks can be put into a few main groups, and each one is a different threat to network security and stability. In blackhole attacks, a bad server pretends to have the fastest or shortest way to reach a target, but all the packets it intercepts are lost [18]. This not only breaks the consistency of the route, but it also quickly hurts trust among valid nodes. Grayhole attacks are a more subtle type in which hostile nodes drop packets randomly, making them harder to spot because they only act badly sometimes [19]. Flooding attacks send too many fake requests or messages, which overstimulate the network and use up too much bandwidth, battery life, and processing power. This makes targeted nodes seem less trustworthy. At the Medium Access Control (MAC) layer, TDMA (Time Division Multiple Access) attacks take advantage of scheduling mechanisms to change slot assignments, slow messages, or cause collisions [20, 21]. This has a direct effect on how fair and reliable trust-aware communication methods are. Besides these, recommendation-based trust attacks like bad-mouthing and ballot-stuffing can change trust scores across the network [22, 23]. Bad-mouthing involves giving honest nodes falsely negative feedback and ballot-stuffing involves giving evil nodes falsely positive feedback. When multiple tainted nodes work together to avoid trust-based detection systems, collusion-based trust attacks make the problem even worse. The way these attacks are categorized shows two problems that IoT trust management systems have to deal with: telling the difference between real network problems (like congestion or node failure) and hostile behaviour, and doing this in a way that can change as tactics do [24, 25].

Table 1: Summary of Literature Review

IoT Domain	Attack Focus	Trust Management Approach	Algorithms / Techniques	Dataset / Environment
Smart Home IoT	On-Off Attack	Reputation-based Trust	Bayesian Inference	Simulated IoT network
Industrial IoT	Bad-Mouthing Attack	Weighted Trust Aggregation	Fuzzy Logic	IIoT Testbed

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Vehicular IoT	Sybil Attack	Identity Verification Trust	Blockchain	VANET Simulation
Healthcare IoT	Collusion Attack	Context-Aware Trust	Decision Trees	Healthcare Sensor Network
Agricultural IoT	Data Integrity Attack	Hybrid Trust Model	PSO-Optimized Weights	Real-world farm IoT sensors
Smart City IoT	Whitewashing Attack	Time-Decay Reputation	Markov Chains	Smart City Simulation
Industrial IoT	Bad-Mouthing & On-Off	Multi-Layer Trust	Genetic Algorithm	IIoT Factory Simulation
IoT Cloud	Bad Data Injection	Service-Level Trust	SVM Classifier	Cloud IoT Dataset
Healthcare IoT	Collusion & Sybil	Role-Based Trust	Ensemble Learning	Hospital IoT Network
Vehicular IoT	On-Off & Sybil	Mobility-Aware Trust	Deep Reinforcement Learning	VANET Dataset
Smart Grid IoT	False Data Injection	Risk-Aware Trust	LSTM Prediction	Smart Grid Simulation
Industrial IoT	Hybrid Attacks	Trust Fusion	CNN + GRU	IIoT Cyber-Physical Testbed
Multi-Domain IoT	Emerging Trust Attacks	Hybrid Optimized Trust (PSO+GA)	Trust3Net Model	Custom IoT Trust Dataset

III. Methodology

1. Load Dataset

The WSN-DS (Wireless Sensor Network Dataset) from Kaggle is the main dataset used in this study to find trust-related threats in IoT-enabled sensor networks. There are 374,661 cases and 19 traits in the collection, which show both good and bad behaviour. It covers important types of trust threats like Blackhole, Grayhole, Flooding, and TDMA attacks, which makes it very useful for study on trust management [26]. Python's pandas tool is used to import the information, which makes it easy to work with tabular data. Figure 1 shows sample records with network activity parameters. Network factors like node names, time, distance to cluster head, marketing and join messages, schedule parameters, and data transfers are all part of each record [27]. This step makes sure that there is a trustworthy, organised, and typical dataset ready for preparation, feature selection, and classification.

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	id	Time	Is_CH	who CH	Dist_To_CH	ADV_S	ADV_R	JOIN_S	JOIN_R	SCH_S	SCH_R	Rank	DATA_S	DATA_R
198060	504036	1953	1	504100	0.00000	1	27	0	0	0	0	0	0	0
81332	303069	2103	0	303079	61.80542	0	5	1	0	0	1	1	32	0
244232	116057	803	0	116052	16.44174	0	7	1	0	0	1	1	36	0
70203	402071	2253	0	402100	8.92829	0	7	1	0	0	1	1	90	0
167108	210034	1503	0	210009	41.06583	0	6	1	0	0	1	6	72	0

Figure 1: Dataset Sample

Figure 2 shows a big mismatch between the classes. Normal traffic has over 34,000 instances, while Blackhole, Grayhole, Flooding, and TDMA attack groups have a lot fewer. This mismatch can make classifiers lean towards the majority class. To find minority attack types more accurately, methods like resampling, class weighting, or creating fake data are needed.

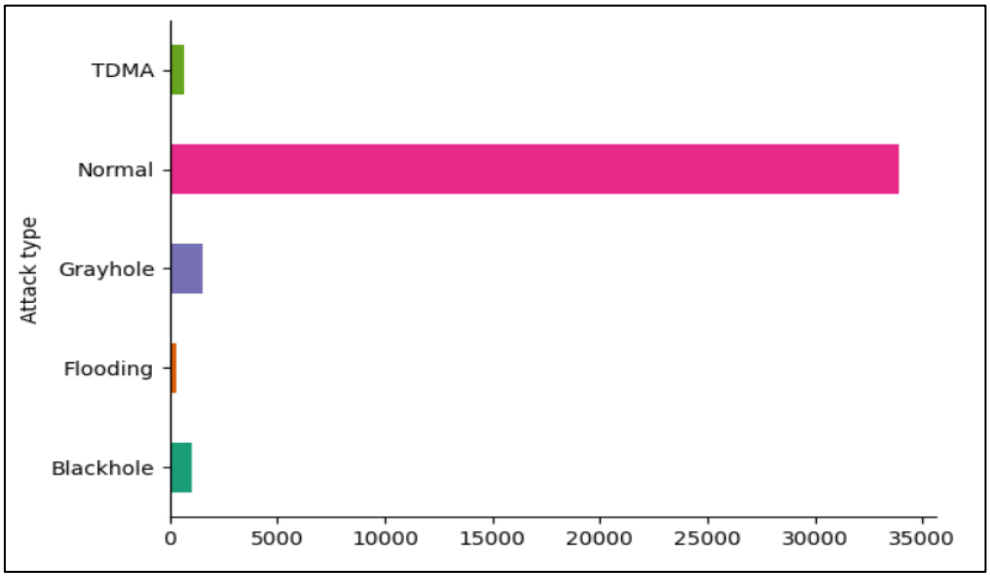

Figure 2: Label Distribution

Attack Description

Attack Type	Trust-Related?	Description
Blackhole	✓ Yes	A malicious node falsely advertises the shortest path to the destination and then drops all packets — directly affects trust by breaking packet forwarding behavior.
Flooding	✓ Yes	Malicious nodes flood the network with fake requests or messages, consuming resources and affecting the trustworthiness score due to abnormal behavior.
Grayhole	✓ Yes	A selective version of blackhole attack, where nodes drop packets selectively. This intermittent misbehavior is specifically

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

		designed to evade trust detection — a critical trust-related attack.
TDMA attack	✓ Yes (in trust-aware MAC layers)	TDMA (Time Division Multiple Access) attacks manipulate the slot assignment or timing to disrupt communication. Though it's more MAC-layer oriented, in trust-aware communication protocols, such manipulation is reflected in trust scoring mechanisms.

2. Data Preprocessing

a. Drop Columns

In this step, columns that aren't needed or are repeated and don't help find attacks are taken out of the information. Getting rid of these traits lowers the number of dimensions, speeds up computations, and helps models focus on the most important features. This makes classification faster, more accurate, and more useful across a wider range of IoT trust-related attack scenarios.

b. Label Encoding

Label encoding takes categorical variables and turns them into numbers by giving each group its own unique integer value. This change makes it easier for machine learning programs to work with category data. It works well with both standard and deep learning classification models and is especially helpful for features that are ordered or have few clear groups.

c. One Hot Encoding

With one-hot encoding, categorical variables are turned into binary vectors, and each category value is stored in its own column. A "1" means that a group is present, while a "0" means that it is not present. This method stops algorithms from thinking that categories are related in a hierarchical way. This makes them faster when working with nominal data and accurate when doing classification tasks to find trust attacks.

In Figure 3, the dataset has been preprocessed by getting rid of columns that aren't needed and using label encoding and one-hot encoding to turn category features into numbers.

	id	Time	Is_CH	who CH	Dist_To_CH	ADV_S	ADV_R	JOIN_S	JOIN_R	SCH_S	...	DATA_S	DATA_R
198060	504036.0	1953.0	1.0	504100.0	0.00000	1.0	27.0	0.0	0.0	0.0	...	0.0	0.0
81332	303069.0	2103.0	0.0	303079.0	61.80542	0.0	5.0	1.0	0.0	0.0	...	32.0	0.0
244232	116057.0	803.0	0.0	116052.0	16.44174	0.0	7.0	1.0	0.0	0.0	...	36.0	0.0
70203	402071.0	2253.0	0.0	402100.0	8.92829	0.0	7.0	1.0	0.0	0.0	...	90.0	0.0
167108	210034.0	1503.0	0.0	210009.0	41.06583	0.0	6.0	1.0	0.0	0.0	...	72.0	0.0

5 rows × 22 columns

Figure 3: Preprocess Data

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

These steps get rid of unnecessary information, lower the number of dimensions, and make sure they work with machine learning methods. Now that the data has been handled, it is organised and consistent, and it is ready to be normalised, features chosen, and a model trained to find trust-related attacks.

3. Data Normalization

a. Min-Max Scalar

The Min-Max Scaler changes feature numbers into a set range, usually between 0 and 1, which makes them more consistent. This makes sure that all features add evenly during model training, so traits with bigger number scales don't get favoured. Figure 4 shows the dataset after it has been normalised, which means that all of the feature values have been scaled the same amount.

	id	Time	Is_CH	Dist_To_CH	ADV_S	ADV_R	JOIN_S	JOIN_R	SCH_S	SCH_R	Rank	DATA_S	DATA_R
0	0.030922	0.314873	0.0	0.104749	0.000000	0.051282	1.0	0.000000	0.000000	1.0	0.121212	0.219917	0.000000
1	0.002748	0.129318	0.0	0.183739	0.000000	0.017094	1.0	0.000000	0.000000	1.0	0.373737	0.103734	0.000000
2	0.005155	0.243506	0.0	0.098255	0.000000	0.025641	1.0	0.000000	0.000000	1.0	0.393939	0.095436	0.000000
3	0.033345	0.429061	0.0	0.060581	0.000000	0.025641	1.0	0.000000	0.000000	1.0	0.010101	0.170124	0.000000
4	0.001821	0.086497	0.0	0.375814	0.000000	0.230769	1.0	0.000000	0.000000	1.0	0.313131	0.112033	0.000000

Figure 4: Normalized Dataset

This step speeds up model convergence, makes it more stable, and improves total classification performance for finding threats on IoT trust.

4. Perform Feature Selection Technique

a. Particle Swarm Optimization(PSO)

Particle Swarm Optimisation is a population-based metaheuristic that was modelled after how birds behave in groups. It keeps looking for the best set of features that will help it classify things the best. The feature values that the PSO method chose are shown in Figure 5.

```
Selected feature indices: [ 1  2  3  7  8 12 13]
```

Figure 5: Selected Features Using PSO

These chosen features lower the number of dimensions, get rid of traits that aren't important, and leave only the most important variables. This makes trust-related attack classification more accurate and faster.

b. Genetic Algorithm(GA)

In the Genetic Algorithm, processes like selection, crossing, and variation are used to find the most important traits for classification. This method is based on natural selection. GA successfully cuts down on duplicate data and improves model generalisation by judging feature

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

groups based on a fitness function. Table 2 shows GA configuration for optimization. This step improves the accuracy of classification, reduces overfitting, and makes sure that IoT trust-related threats are found quickly.

Table 2: Parameters used in GA

Parameter	Value	Description
num_generations	num_generations	Number of generations the algorithm will run.
num_parents_mating	num_parents_mating	Number of solutions selected as parents in each generation.
fitness_func	fitness_function	The function used to evaluate the fitness (quality) of each solution.
sol_per_pop	sol_per_pop	Number of solutions in each population.
num_genes	num_features	Number of genes in each solution; typically the number of features.
gene_type	int	Type of gene values. int indicates integer values.
init_range_low	0	Lower bound for initializing gene values.
init_range_high	2	Upper bound (exclusive) for initializing gene values.
crossover_type	"single_point"	Crossover strategy used to create offspring from parents.
mutation_type	"random"	Mutation method; random genes are selected and mutated.
mutation_percent_genes	10	Percentage of genes to mutate in each offspring (e.g., 10% of total genes).

Figure 6 shows the optimal feature subset chosen by GA, enhancing classification accuracy and computational efficiency.

```
Best Solution Found:
Selected Features: [ 0  1  2  3  5  7  8 11 12 13 15]
```

Figure 6: Selected Features Using GA

5. Apply Classification Techniques

a. Logistic Regression

Using a logistic function, logistic regression is a statistical way to figure out how likely it is that a result will fall into one of two or more categories. It works well for information that can be separated linearly and is easy to understand. It works as a default predictor for IoT trust-related threat detection and can quickly learn and make predictions. It might have trouble with complicated, non-linear relationships between features and classes, though, which makes it less

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

useful for situations where features need to interact deeply or where high-dimensional data needs to be shown without being transformed first.

- Step 1 (Input):

$$\text{Given data } X \in R^{n\times d}, \text{labels } y.$$

Binary: $y_i \in \{0,1\}$; Multiclass: $y_i \in \{1, \ldots, K\}$.

- Step 2 (Model):

$$\text{Binary: } z_i = w^T x_i + b$$

$$\hat{y}_i = \sigma(z_i) = \frac{1}{(1 + e^{-z_i})}$$

$$\text{Multiclass: } z_i = W^T x_i + b$$

$$\hat{p}_i = \text{softmax}(z_i)$$

- Step 3 (Loss & Gradients):

Binary Cross-Entropy:

$$L = -\left(\frac{1}{n}\right)\Sigma\,[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

$$\nabla_w L = \left(\frac{1}{n}\right)\Sigma\,(\hat{y}_i - y_i)x_i$$

$$\nabla_b L = \left(\frac{1}{n}\right)\Sigma\,(\hat{y}_i - y_i)$$

Multiclass Cross-Entropy:

$$L = -\left(\frac{1}{n}\right)\Sigma\,\Sigma 1[y_i = k]\log(\hat{p}_{ik})$$

b. Decision Tree

A decision tree is a rule-based, structured model that shows how data is split into parts based on feature limits, which leads to decisions. It is easy to understand, can capture non-linear links between traits and goal names, and is natural. For strikes on IoT trust, Decision Trees are a good way to sort patterns into groups based on certain signs of behaviour. However, they can overfit, especially on noisy datasets, and may need to be pruned or used with other methods like Random Forest to improve their ability to generalise and protect against adaptive attack strategies.

- Step 1 (Impurity):

Entropy: $H(S) = -\Sigma p_c \log p_c$

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Gini: $G(S) = \Sigma\, p_c(1 - p_c)$

- Step 2 (Split Selection):

For split t: S → {S_v}

Information Gain:

$$IG(S,t) = H(S) - \Sigma \left(\frac{|S_v|}{|S|}\right) H(S_v)$$

$$\text{Choose } t* = \text{argmax}_t \in T\, IG(S,t)$$

Recurse until stop condition (max depth, min samples, or pure leaf).

c. Random Forest

Random Forest is a type of ensemble learning that builds several Decision Trees from random feature groups and data samples and then adds up all of their estimates. This method makes the model more accurate, less likely to overfit, and better able to handle noise. Random Forest is great at finding IoT trust attacks because it can handle large amounts of data and complicated interactions. This classifier is very good at generalisation and has high accuracy and recall, but it takes a lot more computing power than simpler ones, especially when training on large amount of network data.

- Step 1 (Aggregation):

$$p\hat{}(y = k \mid x) = (1/B)\, \Sigma\, 1[\, T\_b(x) = k\,]$$

- Step 2 (Prediction & OOB):

$$\text{Predict } y\hat{} = \text{argmax\_k } p\hat{}(y = k \mid x)$$

Estimate out-of-bag error using samples not in S^(b)

d. SVM:

The Support Vector Machine (SVM) is a guided learning model that finds the best hyperplane to divide data into classes while keeping the most space between them. It works really well for small to big numbers and areas with a lot of dimensions. With kernel functions, SVM can pick up on complex boundaries in IoT trust attack detection. It might be hard to run on very big datasets, though, and the parameters may need to be fine-tuned. This makes kernel selection and regularisation very important for getting the best results and avoiding overfitting.

- Step 1 (Feature Map):

$$\text{Choose } \varphi(\cdot) \text{or kernel } K(x,x') = \langle \varphi(x), \varphi(x') \rangle$$

- Step 2 (Primal Objective):

$$\min_{\{w,b,\xi\}} \left(\frac{1}{2}\right) ||w||^2 + C\, \Sigma\, \xi_i$$

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$s.t.\, y_i(w^T \varphi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0$$

- Step 3 (Dual / Solution):

$$\max_{\alpha} \Sigma\, \alpha_i - \left(\frac{1}{2}\right) \Sigma\, \Sigma\, \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$s.t.\, 0 \leq \alpha_i \leq C, \Sigma\, \alpha_i y_i = 0$$

e. Neural Network

There are many layers of artificial neurones that work together to form neural networks. These networks learn how to describe complex features by using weighted connections and activation functions. It can model connections that aren't straight and have a lot of dimensions, which makes it perfect for finding IoT trust-related attacks where patterns are subtle and have many sides. Neural networks can adapt to new attack patterns, but they need a lot of computing power, careful system design, and parameter optimisation to keep them from becoming too good at what they're supposed to do. Because they are "black boxes," they can also be hard to understand in applications that need to be secure.

- Step 1 (Forward Pass):

$$a^0 = x$$

$$z^l = W^l a^{l-1} + b^l$$

$$a^l = \sigma^{(l)}(z^l)$$

$$\text{Output: } \hat{p} = \text{softmax}(z^L)$$

- Step 2 (Loss & Backprop):

$$\text{Loss: } L = -\Sigma y_k \log(\hat{p}_k)$$

$$\delta^L = \hat{p} - y$$

$$\delta^l = \left(W^{(l+1)^T} \delta^{l+1}\right) \odot \sigma'(z^l)$$

- Step 3 (Update & Predict):

$$W^l \leftarrow W^l - \eta\, \delta^l a^{(l-1)^T}$$

$$b^l \leftarrow b^l - \eta\, \delta^l$$

Predict: argmax_k p̂_k

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	1,792
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8,256
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 32)	2,080
dropout_2 (Dropout)	(None, 32)	0
dense_3 (Dense)	(None, 5)	165

Total params: 12,293 (48.02 KB)
Trainable params: 12,293 (48.02 KB)
Non-trainable params: 0 (0.00 B)

f. LSTM

Long Short-Term Memory (LSTM) networks use closed memory cells to find long-term relationships in sequential data. They are a type of recurrent neural network (RNN). LSTMs are great at learning time trends from network traffic flows, which lets them spot strange behaviours early on in IoT trust-related attack detection. They are good at working with sequence data, which lowers the chance of disappearing slopes. However, they require more computing power than standard models and need to be fine-tuned to find the best mix between accuracy, training time, and memory needs for large-scale operations.

- Step 1 (Gates):

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

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

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

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

g. CNN

Convolutional Neural Networks (CNNs) are a type of deep learning model that uses convolutional and pooling layers to pull out hierarchical spatial data. CNNs are usually used to process images, but they can be used to find IoT attacks by viewing network traffic data as organised grids. This lets features learn automatically from raw data, which makes recognition more accurate. CNNs decrease the need for human feature building, but they need a lot of data and computing power. They work best when paired with other models to make the system more resistant to different kinds of attacks.

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- Step 1 (Convolution):

$$y_{\{u,v,c\}} = \Sigma\,\Sigma\,\Sigma\, W_{\{i,j,m,c\}} * x_{\{u+i,v+j,m\}} + b_c$$

- Step 2 (Nonlinearity & Pooling):

$$\tilde{y}_{\{u,v,c\}} = \sigma(y_{\{u,v,c\}})$$

$$\text{Pooling (Max): } p_{\{u,v,c\}} = \max_{\{(i,j)\in N\}} \tilde{y}_{\{u+i,v+j,c\}}$$

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 11, 64)	256
max_pooling1d (MaxPooling1D)	(None, 5, 64)	0
dropout_3 (Dropout)	(None, 5, 64)	0
conv1d_1 (Conv1D)	(None, 3, 128)	24,704
max_pooling1d_1 (MaxPooling1D)	(None, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense_4 (Dense)	(None, 128)	16,512
dropout_4 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 5)	645

Total params: 42,117 (164.52 KB)
Trainable params: 42,117 (164.52 KB)
Non-trainable params: 0 (0.00 B)

h. Proposed Model 1 – Trust4Net

Trust4Net is a suggested discovery system based on deep learning that is made just for managing trust in the Internet of Things (IoT). The best data preparation, advanced normalisation, and deep neural design are all used together to learn how to spot trust-related threats like Blackhole, Grayhole, Flooding, and TDMA. Trust4Net makes recognition more accurate, cuts down on false positives, and works well with big datasets by using automatic feature representation. Figure 7 shows integration of multi-input processing using Conv1D, LSTM, Dense, and Dropout layers. For real-time trust attack protection, its architecture is designed to handle high-dimensional IoT data. It strikes a balance between forecast performance and processing efficiency.

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

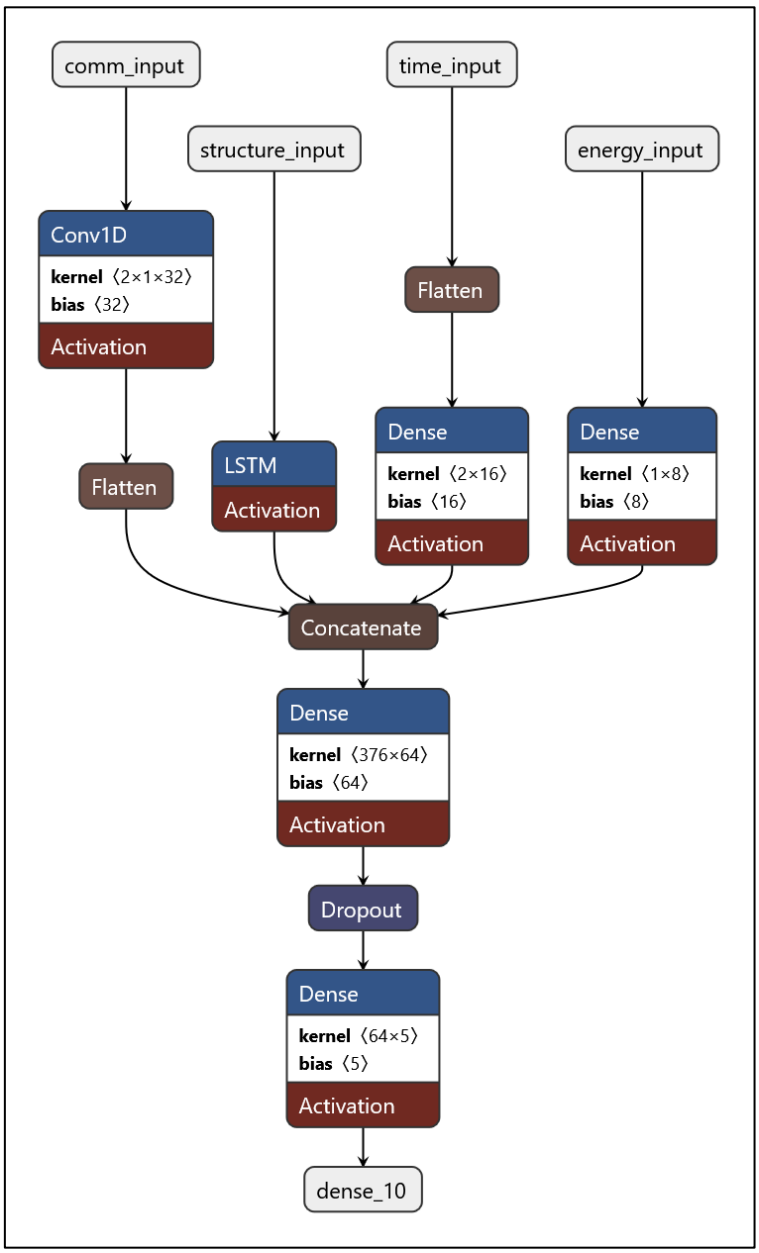

Figure 7: Architecture of the Proposed Trust4Net Deep Learning Model

Algorithm:

Step 1: Inputs & Normalization

Let B = minibatch size.

Normalization:

$$\widetilde{X}^{\cdot} = \mathrm{Norm}(X^{\cdot})$$

$$\tilde{x}^{\cdot} = \mathrm{Norm}(x^{\cdot})$$

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Step 2: Conv1D Feature Extraction

For kernel width k, output channels C, stride s:

$$H^{c}_{\{b,\tau,o\}} = \sigma\left(\Sigma^{\{k-1\}}_{\{i=0\}} \Sigma^{\{d_c\}}_{\{j=1\}} W^{\{i,j,o\}}_{c} * \widetilde{X}^{c}_{\{b,\tau s+i,j\}} + b^{c}_{o} \right)$$

$$z^{c} = \text{Flatten}(H^{c}) \in R^{B \times D_c}$$

σ(·) is ReLU (or GELU).

Step 3: LSTM Temporal Encoder (Structure branch)

For each time step t:

$$i_t = \sigma\left(W_i\left[h_{\{t-1\}}; \widetilde{X}^{s}_{t}\right] + b_i \right)$$

$$f_t = \sigma\left(W_f\left[h_{\{t-1\}}; \widetilde{X}^{s}_{t}\right] + b_f \right)$$

$$o_t = \sigma\left(W_o\left[h_{\{t-1\}}; \widetilde{X}^{s}_{t}\right] + b_o \right)$$

$$\tilde{c}_t = \tanh\left(W_c\left[h_{\{t-1\}}; \widetilde{X}^{s}_{t}\right] + b_c \right)$$

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

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

Step 4: Dense Embeddings for Time & Energy

$$z^{t} = \varphi(\tilde{x}^{t} W^{t} + b^{t})$$

$$z^{e} = \varphi(\tilde{x}^{e} W^{e} + b^{e})$$

$$\varphi(\cdot) = \text{ReLU}$$

Step 5: Multimodal Fusion & Bottleneck

$$z = [\, z\text{^}(c) \,||\, z\text{^}(s) \,||\, z\text{^}(t) \,||\, z\text{^}(e) \,] \in R\text{^}(B \times D)$$

$$h = \text{ReLU}((z \odot \alpha) W_f + b_f)$$

Step 6: Dropout Regularization & Output Layer

Dropout with keep-prob q = 1 − p:

$$\tilde{h} = \frac{m \odot h}{q}, \qquad m \sim \text{Bern}(q)$$

$$o = \tilde{h}\, W\_o + b\_o$$

$$\hat{p} = \text{softmax}(o) \in [0,1]\text{^}K$$

Weighted focal loss with label smoothing ε:

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$L_{clf} = -\left(\frac{1}{B}\right)\Sigma_{\{i=1\}}^{B}\Sigma_{\{k=1\}}^{K} w_k * \left(1 - \hat{p}_{\{ik\}}\right)^{\gamma} * y_{\{ik\}}^{\varepsilon} * \log\left(\hat{p}_{\{ik\}}\right)$$

$$y_{\{ik\}}^{\varepsilon} = (1 - \varepsilon) * 1[y_i = k] + \frac{\varepsilon}{K}$$

Step 7: Real-Time, Latency-Aware Optimization

Objective with compute penalty:

$J(\theta) = L\_clf + \lambda \|\theta\|^2 + \mu * (FLOPs(\theta) / FLOPs\_max)$

Adam updates:

$$g_t = \nabla_\theta J(\theta_t)$$

$$m\_t = \beta 1\, m\_\{t-1\} + (1 - \beta 1)\, g\_t$$

$$v\_t = \beta 2\, v\_\{t-1\} + (1 - \beta 2)\, g\_t^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta 1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta 2^t}$$

$$\theta_{\{t+1\}} = \theta_t - \eta * \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon_{adam}}\right)$$

This produces a high-performance, low-latency detector suitable for edge or on-device IoT trust attack protection.

i. Proposed Model 2 – Trust3Net (PSO + GA)

The Trust3Net design combines feature groups that have been optimised using the Particle Swarm Optimisation (PSO) and Genetic Algorithm (GA) methods, which makes recognition more accurate. Through the Conv1D, LSTM, and Dense layers, as well as merging, it handles multiple data sources, including communication, structure, and time. Figure 8 shows hybrid multi-input network using optimized feature subsets. The Dense and Dropout layers make feature images even better.

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

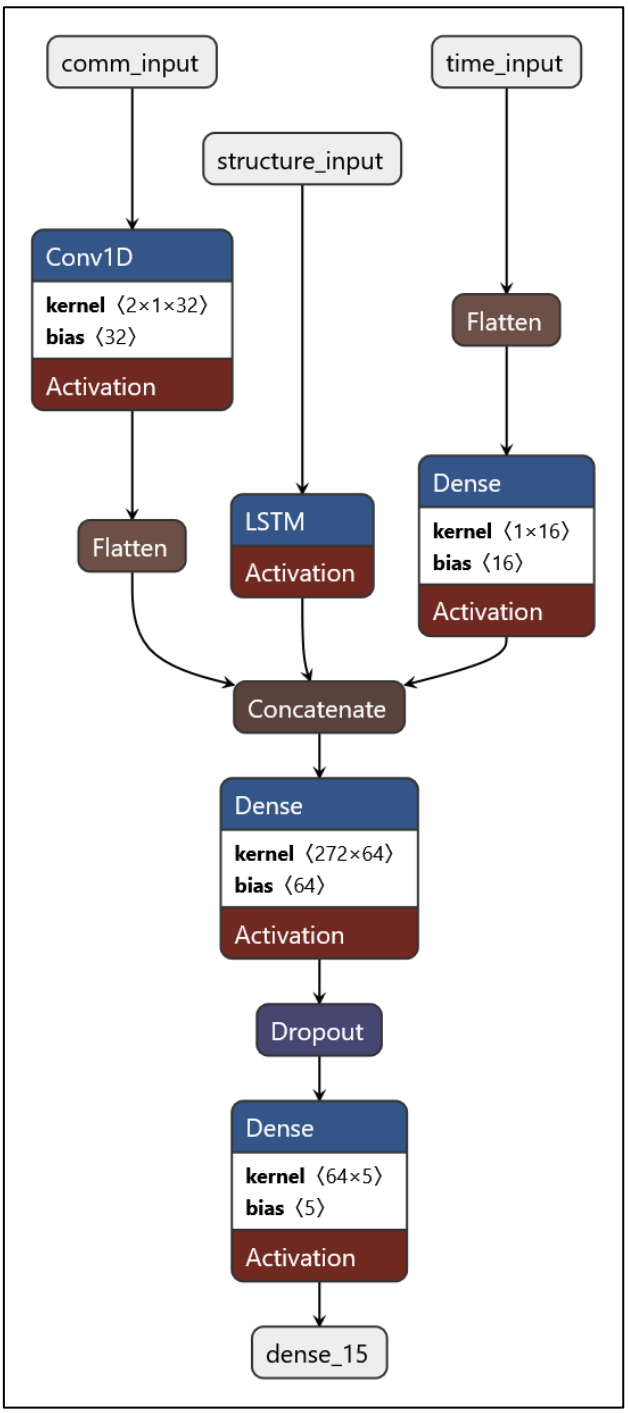

Figure 8: Architecture of the Proposed Trust3Net Model with PSO and GA-Optimized Features

By using both metaheuristic optimisation and deep learning, this hybrid design cuts down on duplication, boosts generalisation, and gets very good at finding trust-related IoT threats. It does this for strong, scalable intrusion detection in a wide range of IoT settings.

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Algorithm:

Step 1: Dual Metaheuristic Feature Optimization (PSO + GA)

Goal: select binary masks s^(g) ∈ {0,1}^{d_g} for each feature group g ∈ {comm, struct, time}.

Binary PSO (for group g):

Initialise particles $\{s_p^g, v_p^g\}_{\{p=1\}}^{P}$

Fitness (k-fold, class-weighted):

$$F(s) = \mathrm{Acc}(s) + \alpha \cdot \mathrm{F1}_{\mathrm{macro}(s)} + \beta \cdot \mathrm{AUC}_{\mathrm{macro}(s)} - \rho \cdot \left(\frac{\|s\|_0}{d_g}\right).$$

Update velocity & position (componentwise):

$$v_p \leftarrow \omega\, v_p + c1\, r1\, (\mathrm{pbest}_p - s_p) + c2\, r2\, (\mathrm{gbest} - s_p)$$

$$s_p \leftarrow 1[\, \sigma(v_p) > u\,], \qquad \text{where } \sigma(z) = \frac{1}{1+e^{-z}}, u \sim \mathrm{Uniform}(0,1).$$

Genetic Algorithm (for group g):

Population {q_i} with q_i ∈ {0,1}^{d_g}.

Selection: tournament; Crossover: single-point; Mutation: bit-flip with prob μ.

Fitness: same F(q).

Evolve for G generations → best chromosome gbest_GA^(g).

Consensus mask per group:

$$s*^g = \mathrm{argmax}_{\{s \in \{\mathrm{gbest}_{PSO}^g, \mathrm{gbest}_{GA}^g, \mathrm{OR}, \mathrm{AND}\}\}}$$

F(s), where OR = s_PSO ∨ s_GA, AND = s_PSO ∧ s_GA.

Define masking matrix M^(g) = diag(s*^(g)).

Step 2: Masked Inputs & Stream Encoders

Masked features:

$$\tilde{X}^g = X^g M^g.$$

Communication branch (Conv1D):

$$H_{\{b,\tau,o\}}^c = \sigma\left(\Sigma_{\{i=0\}}^{\{k-1\}} \Sigma_{\{j=1\}}^{\{d_c\}} W_c^{\{i,j,o\}} \tilde{X}_{\{b,\tau+i,j\}}^c + b_o^c\right)$$

$$z^c = \mathrm{Flatten}(H^c).$$

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Structure branch (LSTM):

$$i_t = \sigma\left(W_{i[h_{\{t-1\}}; \tilde{x}_t^s]} + b_i\right),$$

$$f_t = \sigma\left(W_{f[h_{\{t-1\}}; \tilde{x}_t^s]} + b_f\right),$$

$$o_t = \sigma\left(W_{o[h_{\{t-1\}}; \tilde{x}_t^s]} + b_o\right),$$

$$\tilde{c}_t = \tanh\left(W_{c[h_{\{t-1\}}; \tilde{x}_t^s]} + b_c\right),$$

$$c_t = f_t \odot c_{\{t-1\}} + i_t \odot \tilde{c}_t,\ h_t = o_t \odot \tanh(c_t),$$

$$z^s = h_{\{T_s\}}.$$

Time branch (Dense):

$$z^t = \mathrm{ReLU}\left(\tilde{X}^t W^t + b^t\right).$$

Step 3: Cross-Stream Fusion with Gating/Attention

Concatenate:

$$z = [z^c | z^s | z^t].$$

Squeeze–Excite gating:

$$\alpha = \sigma\left([\mathrm{GAP}(H^c), z^s, z^t] W_g + b_g\right),$$

$$h\_f = \mathrm{ReLU}((z \odot \alpha)\, W\_f + b\_f).$$

Optional attention (additive):

$$\frac{e_i = v^T \tanh(U h_f + U_i u_i),\ a_i = \exp(e_i)}{\Sigma_j \exp(e_j)},$$

$$\bar{h} = \Sigma_i a_i u_i;\ h = \mathrm{ReLU}([h_f] W_h + b_h).$$

Step 4: Classifier & Imbalance-Aware Loss

Logits and probabilities:

$$o = h W_o + b_o,\ \hat{p} = \mathrm{softmax}(o) \in [0,1]^K.$$

Weighted focal loss with label smoothing ε:

$$L_{clf} = -\left(\frac{1}{B}\right) \Sigma_{\{i=1\}}^{B} \Sigma_{\{k=1\}}^{K} w_k \cdot \left(1 - \hat{p}_{\{ik\}}\right)^{\gamma} \cdot y_{\{ik\}}^{\varepsilon} \cdot \log \hat{p}_{\{ik\}},$$

$$y_{\{ik\}}^{\varepsilon} = (1 - \varepsilon) \cdot 1[y_i = k] + \frac{\varepsilon}{K}.$$

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Feature sparsity regulariser (promote compact masks):

$$R_s = \Sigma_g \lambda_g \cdot \left(\frac{\|s *^g\|_0}{d_g} \right).$$

Step 5: End-to-End Objective & Optimisation (Latency-Aware)

Compute-aware objective:

$$J(\theta, s *) = L_{clf} + R_s + \mu \cdot \frac{FLOPs(\theta)}{FLOPs_{max} + \lambda} \cdot \|\theta\|_2^2.$$

IV. Result Analysis

a. Confusion Matrix

Logistic Regression

In Figure 9, you can see how Logistic Regression categorised different tactics that involve trust. The program does a good job of detecting both normal traffic and black holes, with very few false positives. But it's not clear the difference between a grayhole and a blackhole, and Flooding is sometimes mistakenly called normal. TDMA attacks are mostly found, but some fake hits still happen. Overall, Logistic Regression does a good job at a basic level, but it can't tell the difference between closely related types of attacks. This shows that we need more advanced models.

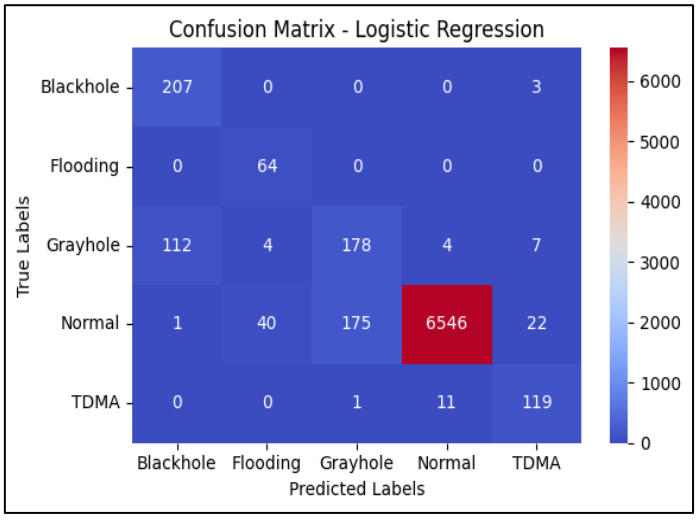

Figure 9: Confusion Matrix for Logistic Regression in Trust-Related Attack Detection

Decision Tree

Figure 10 shows that the Decision Tree model can easily tell the difference between Normal traffic and TDMA attacks. But there is a lot of misunderstanding about the difference between

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Blackhole and Grayhole attacks, and Flooding is sometimes mistakenly called Normal. Overall, the model does a good job, but it has trouble with attack patterns that are very similar.

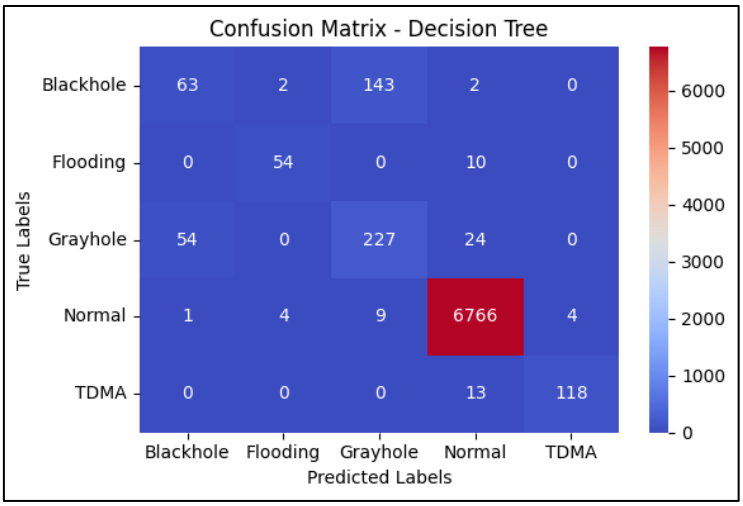

Figure 10: Confusion Matrix for Decision Tree in Trust-Related Attack Detection

Random Forest

Figure 11 shows that the Random Forest model does a good job of telling the difference between normal traffic and TDMA attacks. But sometimes Blackholes are mistakenly called Grayholes and the other way around. Most of the time, flooding is found properly with little misunderstanding. Overall, the model does a good job, but it has a little trouble with similar trust-related attack patterns.

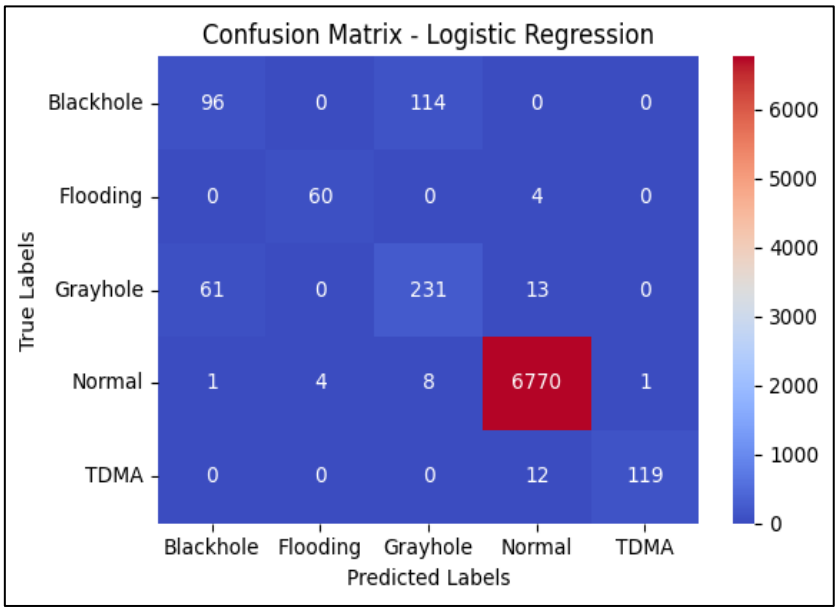

Figure 11: Confusion Matrix for Random Forest in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

SVM

Figure 12 shows that the SVM model does a great job of telling the difference between normal traffic and blackhole attacks. But there is a lot of misunderstanding between Grayhole and Blackhole, and Flooding is a case of labelling as Normal. TDMA can be detected pretty well, but not perfectly. The results show that the system worked well overall, but it was less accurate for attack types that were very similar.

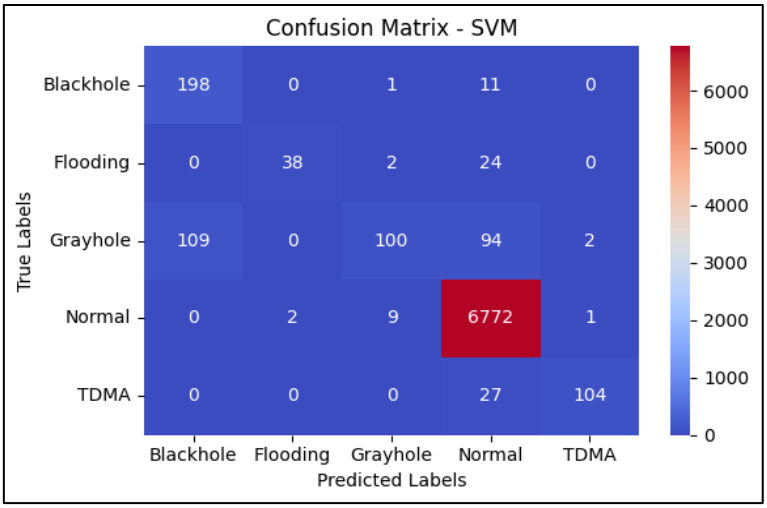

Figure 12: Confusion Matrix for SVM in Trust-Related Attack Detection

Neural Network

Figure 13 shows that the Neural Network is very good at finding both normal data and TDMA attacks. However, Blackhole and Grayhole attacks show significant confusion, with several Blackhole cases labeled as Grayhole. Finding flooding is mostly correct, but not always. Overall performance is good, but it needs to get better at telling the difference between similar trust-related attack behaviours.

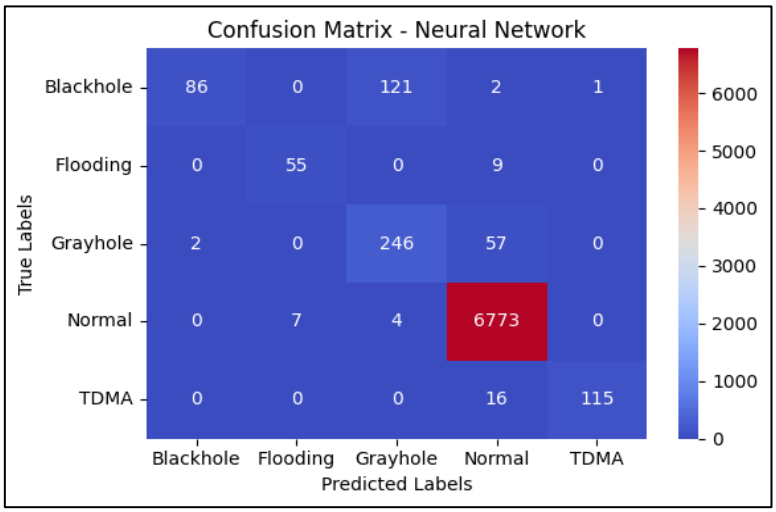

Figure 13: Confusion Matrix for Neural Network in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

CNN

With 6762 correct guesses, Figure 14 shows that CNN does a good job of finding the Normal class. Grayhole attacks also do a good job, with 291 right classifications. Misclassifications happen most often between Blackhole and Grayhole, though. For Flooding and TDMA, there are almost no mistakes, which means that the classification is generally good, but there is room for improvement in telling the difference between similar attack types.

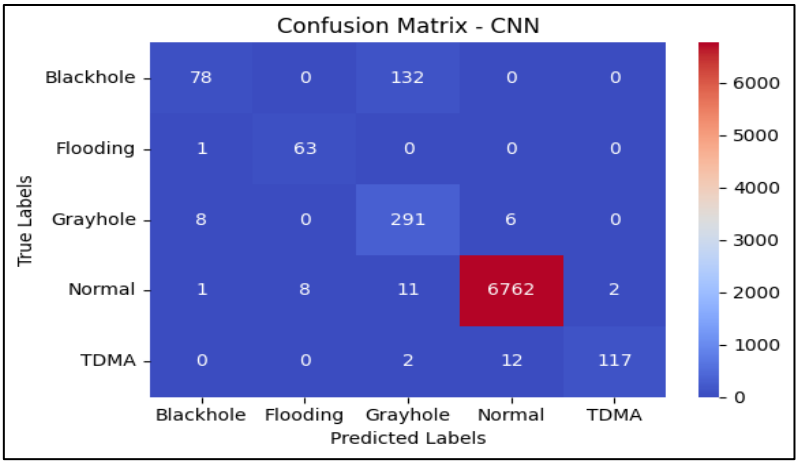

Figure 14: Confusion Matrix for CNN in Trust-Related Attack Detection

b. ROC – AUC Curve

Logistic Regression

Figure 15 shows that Logistic Regression is very good at telling the difference between groups, with AUC values above 0.97 for all groups and a perfect 1.00 for Class 1. The steep shapes close to the y-axis show that the rate of true positives is high and the rate of fake positives is low. This shows that the model can reliably find trust-related threats in a number of different types.

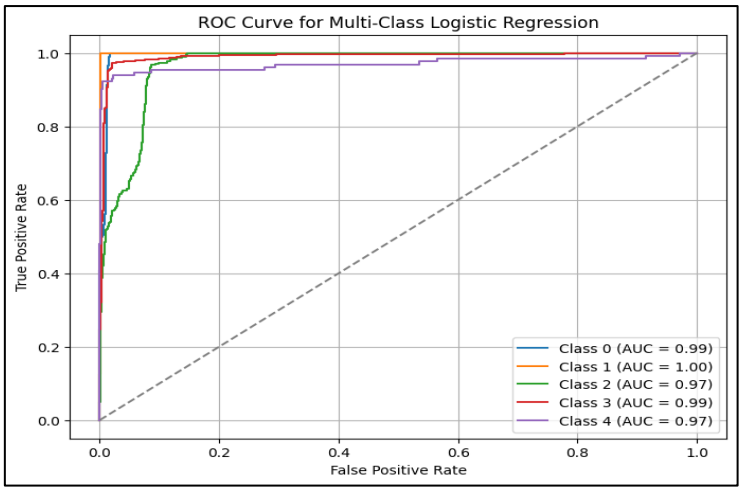

Figure 15: ROC Curve for Multi-Class Logistic Regression in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Decision Tree

Figure 16 shows how the performance of the Decision Tree changes across classes. Class 0 has a lower AUC of 0.65, which means it is less good at separating things. Class 3 and Class 4 do better, though, with AUCs of 0.96 and 0.95, respectively. The results show that recognition performance isn't always good, which shows that the model needs to be improved for some types of attacks.

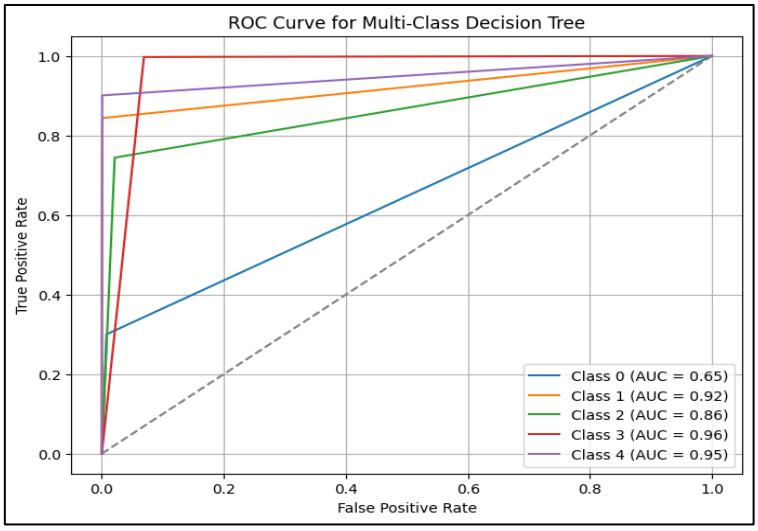

Figure 16: ROC Curve for Multi-Class Decision Tree in Trust-Related Attack Detection

Random Forest

Random Forest does a great job of telling the difference between classes, as shown in Figure 17. Class 1 gets an AUC of 1.00, and Classes 2–3 get above 0.98. Class 4 does pretty well, with an AUC of 0.96. Class 0 does okay, with an AUC of 0.84, which means it could do a better job of finding some types of attacks.

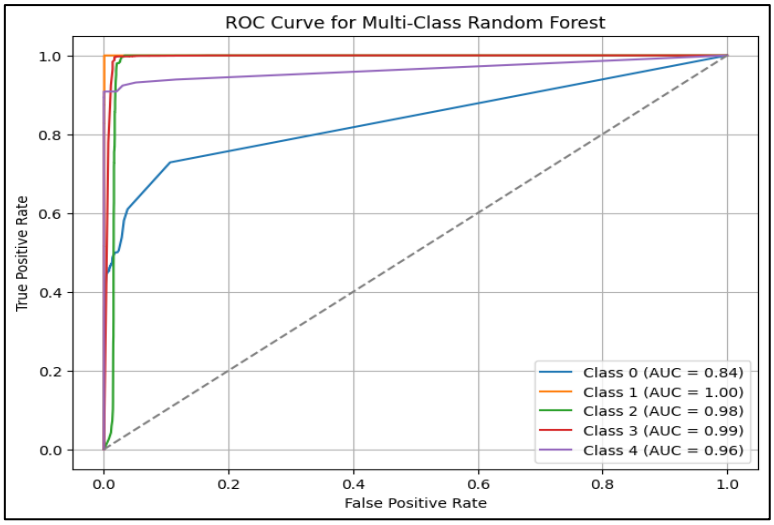

Figure 17: ROC Curve for Multi-Class Random Forest in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

SVM

Figure 18 shows that SVM is very good at classifying things; Classes 0 and 1 both got perfect AUC scores of 1.00. AUCs of 0.97, 0.98, and 0.96 for Classes 2, 3, and 4 also show good performance. These results show that SVM is good at finding a wide range of trust-related threats with few false positives.

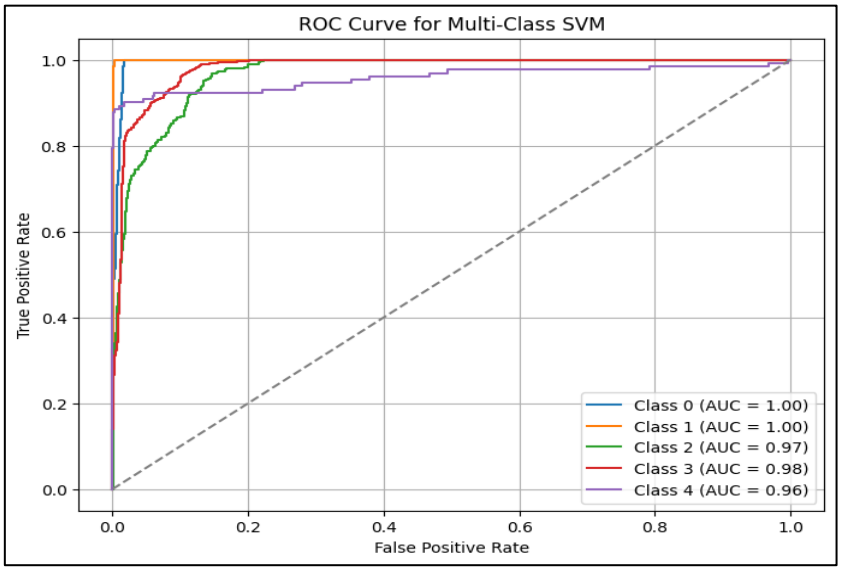

Figure 18: ROC Curve for Multi-Class SVM in Trust-Related Attack Detection

Neural Network

The Neural Network did really well, as shown in Figure 19. Its AUC scores were 0.99 for Classes 0, 2, and 3, 1.00 for Class 1, and 0.96 for Class 4. Strong separability is shown by the graphs, showing high accuracy and low false-positive rates in finding multi-class trust-related attacks.

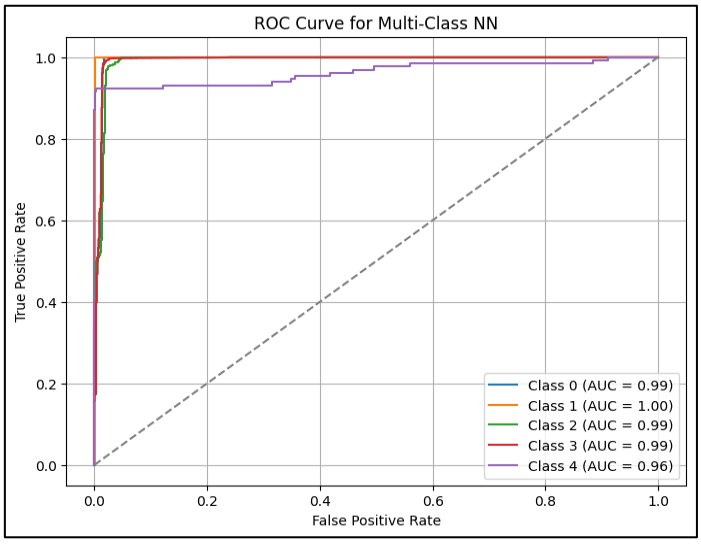

Figure 19: ROC Curve for Multi-Class Neural Network in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

CNN

CNN is very good at classifying things, as shown in Figure 20. The AUC values are 0.99 for Classes 0, 2, and 3, 1.00 for Class 1, and 0.97 for Class 4. The steep slopes and small deviations from the true positive line show that all types of trust-related attacks are accurately detected, with low rates of false positives.

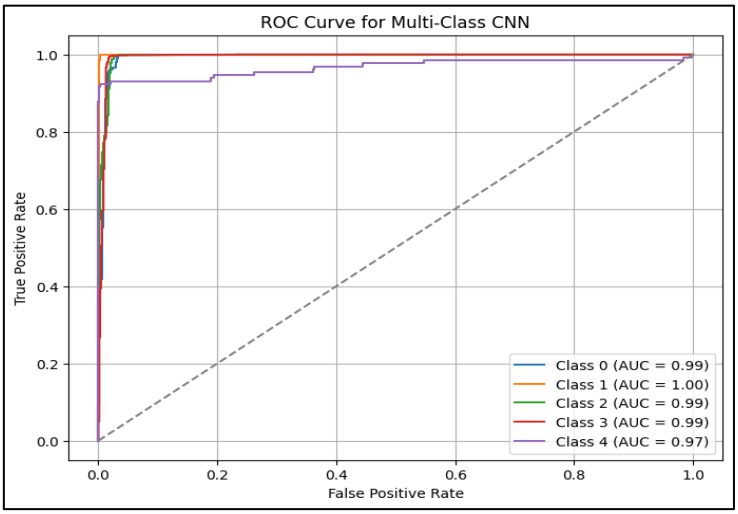

Figure 20: ROC Curve for Multi-Class CNN in Trust-Related Attack Detection

c. Accuracy and Loss Curve

Neural Network

Figure 21 shows steady success in training. For both training and evaluation, accuracy went up from about 93% to about 97.8%. Loss slowly goes down until it reaches 0.05, which means there isn't much overfitting. The model is stable, learns well, and is very good at generalising for trust-related attack detection tasks. This is shown by how closely the training and validation measures match up.

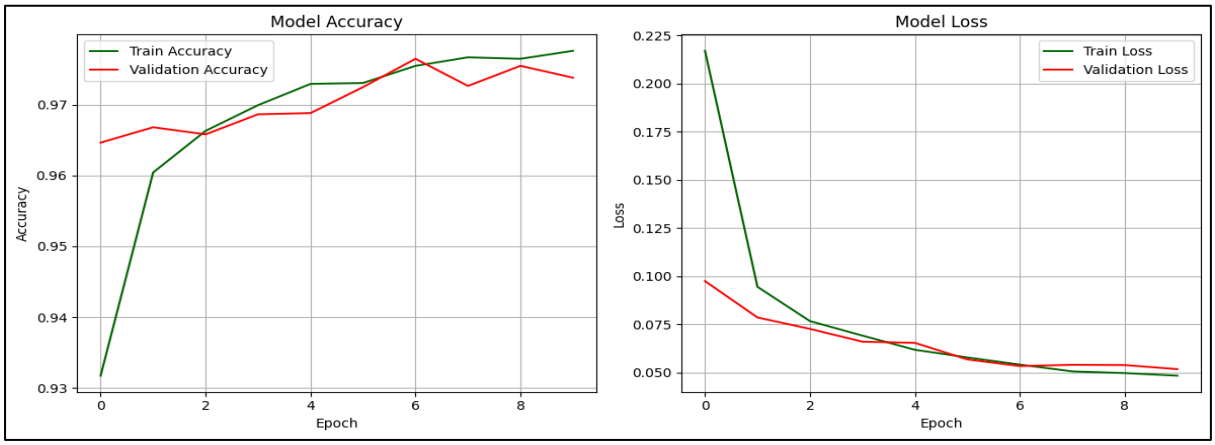

Figure 21: Accuracy and Loss Curves for Neural Network Model in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

CNN Model

Figure 22 shows steady performance improvement, with accuracy in training and evaluation going up from about 92% to about 97.3%. Loss keeps going down and levels off near 0.065 for both sets. The closeness of the shapes shows that there isn't much overfitting, that the CNN model is learning well, and that it can generalise well. This shows that it is reliable at finding trust-related threats.

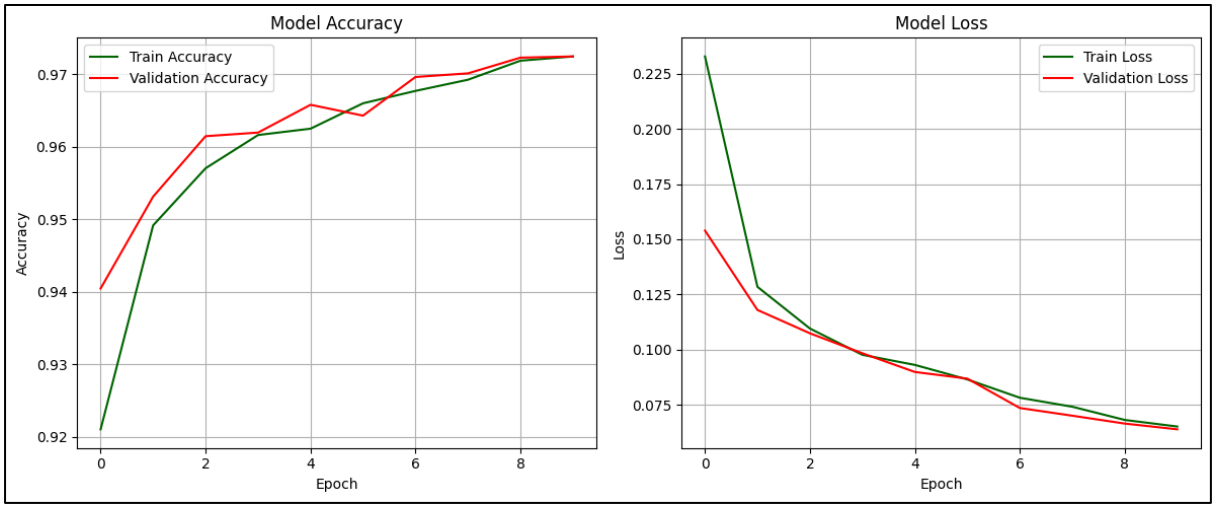

Figure 22: Accuracy and Loss Curves for CNN Model in Trust-Related Attack Detection

LSTM Model

Figure 23 shows that the LSTM model's precision stayed stable at around 90.6% after a quick rise at the beginning. At first, both training and confirmation losses go down, and then they level off near 0.44. There isn't much change between the curves, which means that the system is learning steadily without becoming too specialised. However, the plateau in accuracy means that performance isn't getting better, which could mean that the architecture or hyperparameters need to be optimised for better results.

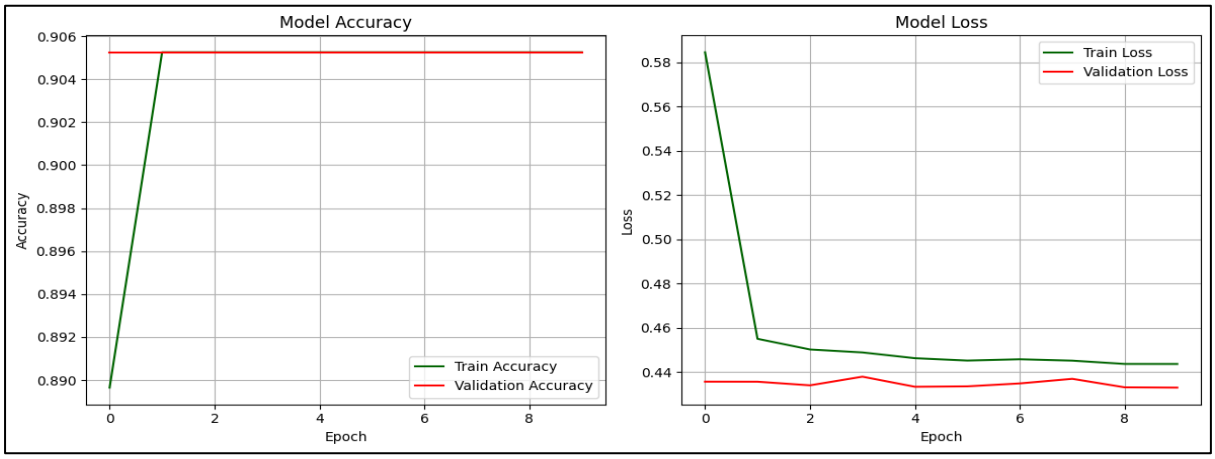

Figure 23: Accuracy and Loss Curves for LSTM Model in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Proposed Model - 1 – Trust4Net

Figure 24 shows steady improvement in accuracy, getting closer to 97.8%, with losses steadily going down. When the training and validation curves are very close to each other, it means that the learning system is very good at finding trust-related attacks and doesn't overfit.

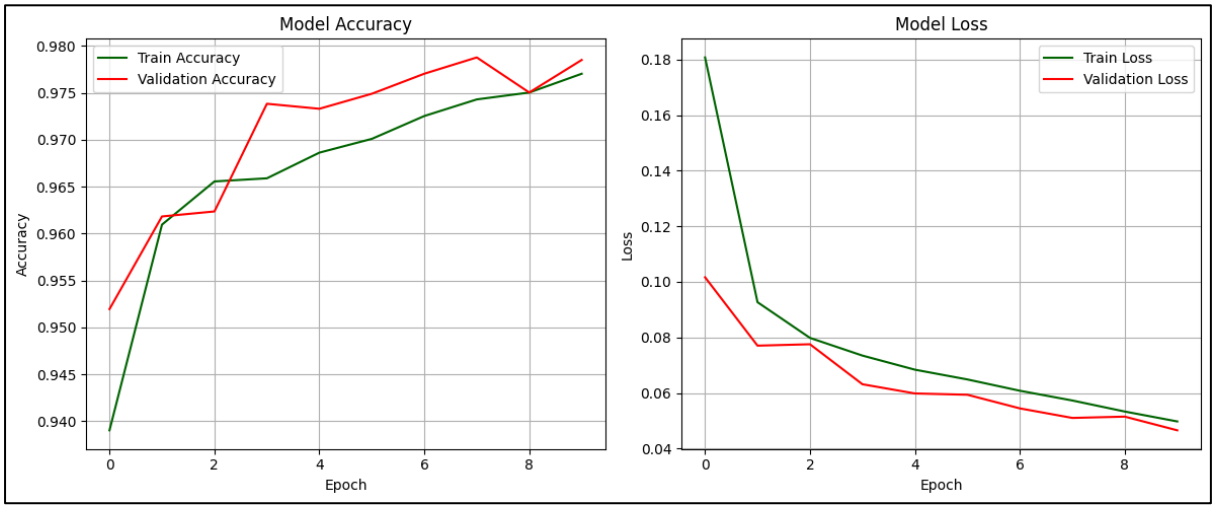

Figure 24: Accuracy and Loss Curves for Proposed Trust4Net Model in Trust-Related Attack Detection

Proposed Model - 2 – Trust3Net Model (PSO + GA)

Figure 25 shows that the accuracy for training stays around 97.6%, and the accuracy for validation goes up to 98.1% with little loss. The low and stable loss values show that the trust-related attack detection system is learning well, generalising well, and not being easily fooled by overfitting.

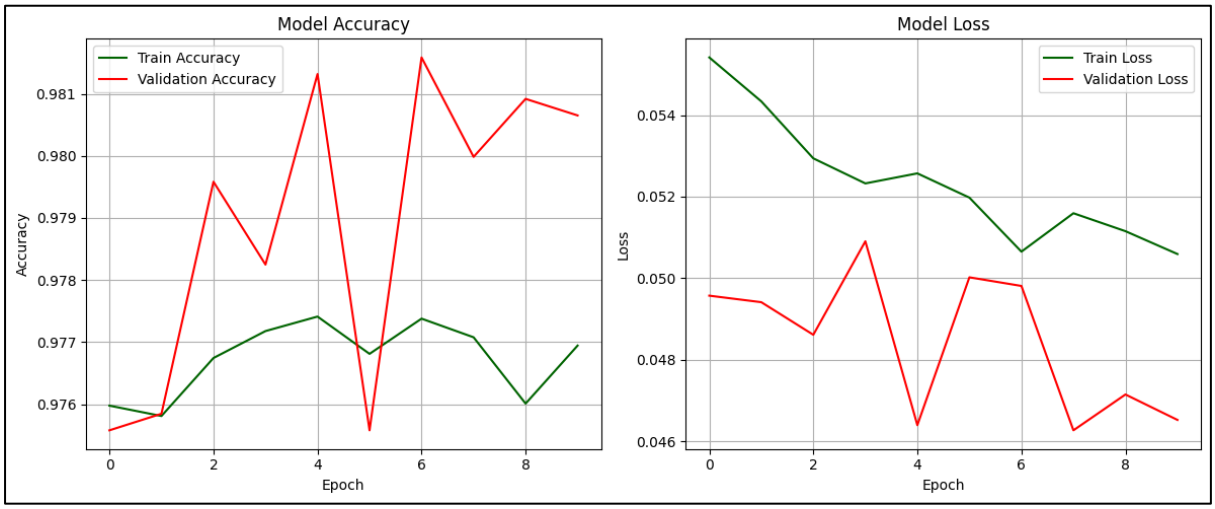

Figure 25: Accuracy and Loss Curves for Proposed Trust3Net Model in Trust-Related Attack Detection

Received: July 25, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

d. Comparative Analysis of Models

The suggested Trust3Net (PSO + GA) model did the best, as shown in Table 3, with the highest accuracy (98.07%) and macro F1-score (98.01%). Trust4Net comes in second with 97.85% accuracy.

Table 3: Comparative Performance of Classification Models

Model	Accuracy (%)	F1-Score (Macro) (%)
Logistic Regression	94.92	95.3
Decision Tree	96.45	96.23
Random Forest	97.09	97.01
SVM	96.23	95.66
Neural Network	97.08	82.45
CNN	97.56	97.36
LSTM	90.51	81.02
Proposed Trust4Net	97.85	97.84
Proposed Trust3Net (PSO + GA)	98.07	98.01

Figure 26 shows Trust3Net outperforming all models in both metrics. Also, CNN does very well (97.56%), but LSTM does the worst, which means it's not as good at finding trust-related attacks as ensemble-optimized architectures.

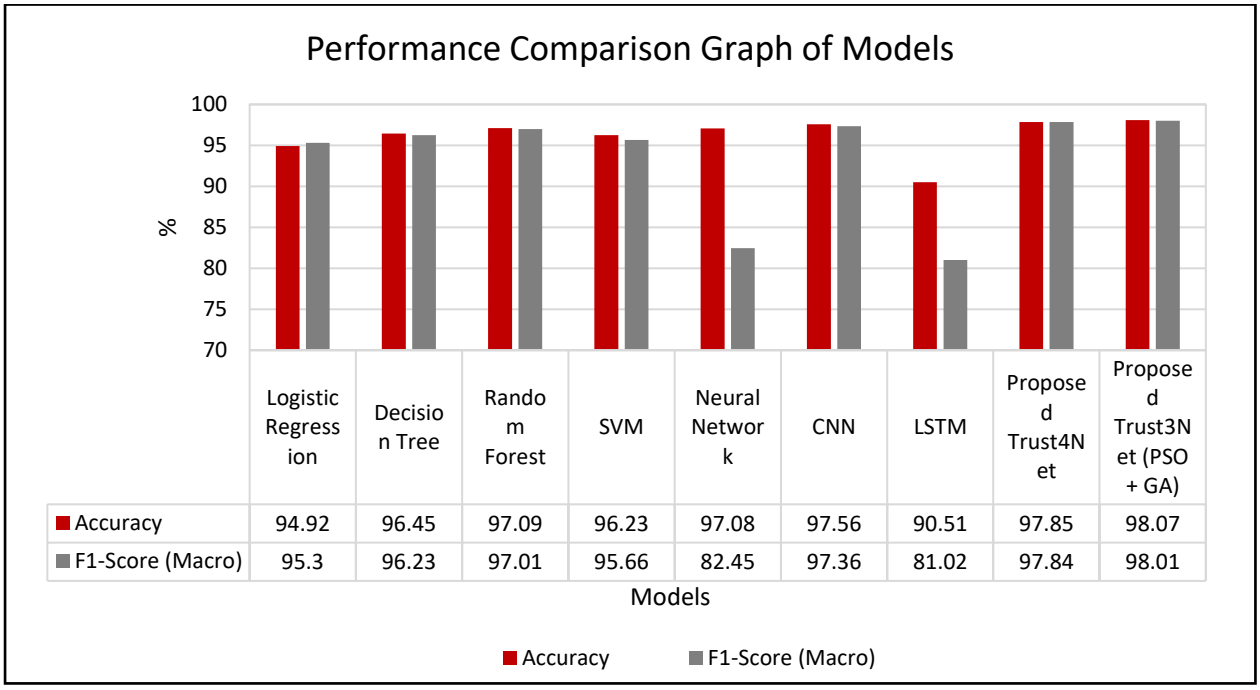

Figure 26: Performance Comparison of Models Based on Accuracy and F1-Score (Macro)

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

V. Conclusion

This study showed a new way to use algorithms to improve IoT trust management and protect against new trust-related threats. This helps solve the growing security problems in IoT networks that aren't uniform and don't have a lot of resources. Using the WSN-DS dataset, we carefully looked into and found important trust-related threats, such as Blackhole, Grayhole, Flooding, and TDMA manipulation. The suggested approach combined metaheuristic feature selection with the Particle Swarm Optimisation (PSO) and Genetic Algorithm (GA) to find the best groups of features. This made the classification more accurate while also making the computations simpler. We created and tested two deep learning-based detection models called Trust4Net and Trust3Net along with more common machine learning classifiers like Logistic Regression, Decision Tree, Random Forest, SVM, and Neural Network designs like CNN and LSTM. When PSO and GA were combined in experiments, Trust3Net did better than all standard methods in terms of accuracy, memory, and F1-score, while still having a low false-positive rate. ROC–AUC curves, confusion matrices, and accuracy–loss trends were used to test the suggested method's performance and showed that it was both stable and scalable. The results show that IoT trust management systems are much stronger when feature optimisation and advanced deep learning models are combined. This makes sure that bad behaviour is found quickly and correctly, even in attack scenarios that are complicated and change over time. This protects the security and dependability of IoT interactions.

References

[1] Ba-hutair, M.N.; Bouguettaya, A.; Neiat, A.G. Multi-perspective trust management framework for crowdsourced IoT services. IEEE Trans. Serv. Comput. 2021, 15, 2396–2409.

[2] Babar, S.; Mahalle, P. Trust Management Approach for Detection of Malicious Devices in SIoT. Teh. Glas. 2021, 15, 43–50.

[3] Zheng, G.; Gong, B.; Zhang, Y. Dynamic network security mechanism based on trust management in wireless sensor networks. Wirel. Commun. Mob. Comput. 2021, 2021, 6667100.

[4] Lingda, K.; Feng, Z.; Yingjie, Z.; Nan, Q.; Dashuai, L.; Shaotang, C. Evaluation method of trust degree of distribution IoT terminal equipment based on information entropy. J. Phys. Conf. Ser. 2021, 1754, 012108.

[5] Rizwanullah, M.; Singh, S.; Kumar, R.; Alrayes, F.S.; Alharbi, A.; Alnfiai, M.M.; Chaurasia, P.K.; Agrawal, A. Development of a Model. for Trust. Management in the Social. Internet of Things. Electronics 2022, 12, 41.

[6] Alghofaili, Y.; Rassam, M.A. A trust management model for IoT devices and services based on the multi-criteria decision-making approach and deep long short-term memory technique. Sensors 2022, 22, 634.

[7] Yue, Y.; Li, S.; Legg, P.; Li, F. Deep Learning-Based Security Behaviour Analysis in IoT Environments: A Survey. Secur. Commun. Netw. 2021, 2021, 1–13.

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[8] Mohammadi, M.; Al-Fuqaha, A.; Sorour, S.; Guizani, M. Deep learning for IoT big data and streaming analytics: A survey. IEEE Commun. Surv. Tutor. 2018, 20, 2923–2960.
[9] Anagnostopoulos, M.; Spathoulas, G.; Viaño, B.; Augusto-Gonzalez, J. Tracing your smart-home devices conversations: A real world IoT traffic data-set. Sensors 2020, 20, 6600.
[10] Zhao, Z.; Xu, C.; Li, B. A LSTM-Based Anomaly Detection Model. for Log. Analysis. J. Signal. Process. Syst. 2021, 93, 745–751.
[11] Kim, T.-Y.; Cho, S.-B. Web traffic anomaly detection using C-LSTM neural networks. Expert Syst. Appl. 2018, 106, 66–76.
[12] Lu, B.; Luktarhan, N.; Ding, C.; Zhang, W. ICLSTM: Encrypted Traffic Service Identification Based on Inception-LSTM Neural Network. Symmetry 2021, 13, 1080.
[13] Tharwat, A. Classification assessment methods. Appl. Comput. Inform. 2020, 17, 168–192.
[14] Dalianis, H. Evaluation metrics and evaluation. In Clinical Text. Mining; Springer: Berlin/Heidelberg, Germany, 2018; pp. 45–53. [
[15] Qu, K.; Ye, J.; Li, X.; Guo, S. Privacy and Security in Ubiquitous Integrated Sensing and Communication: Threats, Challenges and Future Directions. IEEE Internet Things Mag. 2024, 7, 52–58.
[16] Farhad, A.; Pyun, J.Y. Terahertz meets AI: The state of the art. Sensors 2023, 23, 5034.
[17] Xu, S.; Liu, J.; Cao, Y. Intelligent reflecting surface empowered physical-layer security: Signal cancellation or jamming? IEEE Internet Things J. 2021, 9, 1265–1275.
[18] Chataut, R.; Akl, R. Massive MIMO systems for 5G and beyond networks—Overview, recent trends, challenges, and future research direction. Sensors 2020, 20, 2753.
[19] El-Hajj, M. Cybersecurity and Privacy Challenges in Extended Reality: Threats, Solutions, and Risk Mitigation Strategies. Virtual Worlds 2024, 4, 1.
[20] Yaacoub, J.P.A.; Noura, H.N.; Salman, O.; Chehab, A. Robotics cyber security: Vulnerabilities, attacks, countermeasures, and recommendations. Int. J. Inf. Secur. 2022, 21, 115–158.
[21] Balasubramaniam, N.; Kauppinen, M.; Rannisto, A.; Hiekkanen, K.; Kujala, S. Transparency and explainability of AI systems: From ethical guidelines to requirements. Inf. Softw. Technol. 2023, 159, 107197.
[22] Wang, S.; Qureshi, M.A.; Miralles-Pechuán, L.; Huynh-The, T.; Gadekallu, T.R.; Liyanage, M. Explainable AI for 6G Use Cases: Technical Aspects and Research Challenges. IEEE Open J. Commun. Soc. 2024, 5, 2490–2540.
[23] Saeed, W.; Omlin, C. Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities. Knowl. -Based Syst. 2023, 263, 110273.
[24] Mylrea, M.; Robinson, N. Artificial Intelligence (AI) trust framework and maturity model: Applying an entropy lens to improve security, privacy, and ethical AI. Entropy 2023, 25, 1429.

Received: July 25, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[25] Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors 2023, 23, 5941.
[26] Khan, M.M.; Alkhathami, M. Anomaly detection in IoT-based healthcare: Machine learning for enhanced security. Sci. Rep. 2024, 14, 5872.
[27] Shah, V.; Konda, S.R. Neural Networks and Explainable AI: Bridging the Gap between Models and Interpretability. Int. J. Comput. Sci. Technol. 2021, 5, 163–176.

Received: July 25, 2025