Defense Against False Data Injection and Fault Tolerance Optimization for Resource Scheduling in Software-Defined Satellite-Terrestrial Networks Using SFT-LoRA

Huabing Yan
University of Electronic Science and Technology of China, Chengdu, 611731, China
E-mail: yanhuabing@std.uestcedu.cn

Keywords: SFT-LoRA; False data injection; Dynamic fault tolerance; Resource scheduling; LSTM-CNN

Received: September 14, 2025

With the widespread application of Satellite-Terrestrial Integrated Networks (STINs) in key fields such as disaster monitoring, military communications, and the Internet of Things, their resource scheduling systems are facing severe security threats. This paper proposes a Software-Defined Flexible Topology SFT training mechanism that combines spatiotemporal feature analysis and a dynamic fault-tolerant scheduling optimization model. Firstly, in the detection layer, a cross-layer feature extractor based on bidirectional LSTM-CNN is designed, and SFT training mechanism is adopted to realize real-time recognition of injected data; Secondly, at the fault-tolerant layer, build an elastic resource reallocation strategy, predict task demand fluctuations through sliding windows, and dynamically adjust LoRA weight priority and node redundancy; The models were trained and validated on a satellite-terrestrial simulation dataset generated with NS-3 and OpenStack, including 100 nodes, 12 typical attack types, and over 12,000 samples. The training process integrated bidirectional LSTM layers (128 units), CNN filters, and reinforcement learning with PPO for adaptive scheduling. Finally, the experiment is based on NS-3 simulation platform and OpenStack cloud environment, and the injected attack samples cover 12 typical attack modes. Results show that the proposed framework outperforms traditional IDS and a defenseless baseline, achieving a 98.5% FDI detection rate with a false alarm rate below 1.2%. Moreover, the task completion rate after attacks increases to 96.3%, scheduling delay is reduced by 21.7%, and resource utilization is improved by 18.4%. This framework provides an effective solution for resource scheduling security in highly dynamic and weakly connected environments.

Povzetek: Raziskava predstavlja varnostni mehanizem za satelitsko-zemeljska omrežja, ki z uporabo naprednih modelov globokega učenja izboljša zaznavanje napadov ter poveča zanesljivost in učinkovitost razporejanja virov v dinamičnih okoljih.

1 Introduction

In the contemporary digital era, network technology is advancing rapidly and is widely utilized across various domains. The Satellite-Terrestrial Integrated Networks (STINs), as an emerging network architecture with vast potential, is progressively emerging as a crucial medium for information dissemination and sharing [1]. It integrates the strengths of satellite communication, space-based networking, and terrestrial networks to construct a globally encompassing communication framework that ensures seamless coverage and efficient connectivity. This network provides a robust communication infrastructure and data services for pivotal sectors, including aerospace, maritime exploration, emergency response, and intelligent transportation [2].

As the STINs network expands in scale and penetrates deeper into various critical applications, the security and reliability challenges it confronts are intensifying [3]. Resource scheduling, being the central component of the STINs network, is instrumental in maintaining the network's overall performance, optimizing resource allocation, and enhancing service quality. It is tasked with the rational distribution of computational, storage, and bandwidth resources to cater to the diverse needs of users and services, thereby ensuring the network's efficient and stable operation [4]. Presently, the network resource scheduling of STINs is under threat from numerous vulnerabilities, among which the false data injection attack stands out as one of the most detrimental security risks [5]. Malicious actors may disrupt the normal functioning of the resource scheduling system by tampering with, fabricating, or injecting spurious resource requests and status information. This not only results in irrational resource allocation, depriving legitimate users of timely and effective services, but also leads to severe consequences, such as network congestion and system failure, thereby undermining the reliability and stability of the entire STINs network [6]. Conducting in-depth research on false data injection defense mechanisms within the SFT-LoRA (Software-Defined Flexible Topology with Low-Rank Adaptation) resource scheduling of the STINs network is of paramount urgency. As a novel communication technology, SFT-LoRA offers unique advantages in satellite-space network resource scheduling, such as low power consumption, long-range transmission, and high flexibility, making it widely adopted for the transmission and interaction of resource scheduling information. However, its open and distributed nature also renders it susceptible to false data injection attacks.

Despite the development of various defense technologies in the field of network security, including intrusion detection systems, encryption, and authentication, effective solutions to false data injection in the SFT-LoRA resource scheduling context of the STINs network remain elusive [7]. Existing cross-layer detection approaches for STINs that adopt deep learning or hybrid neural architectures typically achieve FDI detection rates around 90% and false alarm rates exceeding 3%, while traditional defense methods often fail to adapt to the complex topology, dynamic environment, and unique characteristics of SFT-LoRA technology, and are unable to accurately and promptly detect and counter false data injection attacks, thereby ensuring the reliability and security of resource scheduling [8]. Additionally, the satellite network's complex hardware, extensive coverage, and diverse user demands expose it to various unexpected failures and anomalies, such as satellite component malfunctions, ground station damage, and signal interference [9]. These factors can introduce erroneous or inconsistent information into the resource scheduling process, affecting the accuracy of scheduling decisions [10]. Therefore, in addition to establishing a robust false data injection defense system, developing an effective dynamic fault-tolerant mechanism is essential for ensuring the reliability of SFT-LoRA resource scheduling in satellite-space networks.

Dynamic fault-tolerant optimization focuses on real-time monitoring, fault detection and localization, and the rapid and effective processing of fault-tolerant resource scheduling [11]. This enables the system to swiftly adjust scheduling strategies, restore normal resource allocation and service provision in the face of false data injection attacks or other faults, minimize the impact of faults on network performance and user service quality, and enhance the overall robustness and availability of the STINs network [12].

To clarify our research design, this study is guided by the following research questions: (RQ1) Can a hybrid LSTM-CNN detector outperform traditional intrusion detection systems in identifying false data injection attacks in STINs? (RQ2) Does the integration of reinforcement learning-driven scheduling and adaptive redundancy improve system stability and resource utilization under dynamic conditions? (RQ3) Can the combined framework demonstrate scalability in heterogeneous satellite-terrestrial networks with real-time constraints?

This paper introduces an SFT training mechanism that integrates spatiotemporal feature analysis with a dynamic fault-tolerant scheduling optimization model. The goal is to dynamically adjust the network topology through software-defined approaches to enhance the network's fault tolerance and anti-attack capabilities. SFT technology leverages techniques such as Software-Defined Networking (SDN) and Network Function Virtualization (NFV) to achieve flexible configuration and dynamic scheduling of network resources. The primary contributions of this paper are as follows:

(1) 1A cross-layer feature extractor based on bidirectional LSTM-CNN is designed, and SFT training mechanism is adopted to realize real-time recognition of injected data;

(2) Construct an elastic resource reallocation strategy, predict task demand fluctuations through sliding windows, and dynamically adjust LoRA weight priority and node redundancy;

(3) The experiment is based on the NS-3 simulation platform and OpenStack cloud environment, and the injected attack samples cover 12 typical attack modes. The results show that the FDI detection rate of the proposed scheme is 98.5%, and the false alarm rate is lower than 1.2%. Under the dynamic fault-tolerant mechanism, the task completion rate of the system after the attack is increased to 96.3%, the scheduling delay is reduced by 21.7%, and the resource utilization rate is increased by 18.4%.

2 Related theoretical knowledge

2.1 LSTM

The Long Short-Term Memory (LSTM) network, an advanced variant of the recurrent neural network (RNN), is specifically engineered to address the challenges of vanishing and exploding gradients that are prevalent in traditional RNNs when modeling long sequences [13]. Its primary innovation is the incorporation of a gating mechanism, which dynamically modulates the flow of information to effectively capture long-range dependencies within sequential data [14]. This capability endows LSTM with substantial benefits in various time series modeling applications, including natural language processing, time series forecasting, and speech recognition [15].

The fundamental structure of LSTM is built upon memory cells, each of which comprises three distinct gating functions: the forget gate, the input gate, and the output gate. These gates facilitate precise management of the internal state. Initially, the forget gate employs the Sigmoid activation function to produce a weight coefficient within the [0, 1] range, thereby determining the extent to which historical state information is retained. The computational procedure is depicted in Equation (1):

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

Where W_f denotes the weight matrix of the forgetting gate; b_f denotes the bias term; σ denotes the sigmoid activation function, which maps the input between (0, 1) and thus determines the forgetting degree of each memory unit; h_{t-1} denotes the hidden state at the previous moment; x_t denotes the input at the current moment.

Secondly, the input gate serves to selectively incorporate the current input information, thereby facilitating dynamic updates to the memory cell's internal state. This gate comprises two parallel pathways: a sigmoid activation layer and a tanh activation layer. The former generates a set of candidate activation vectors, while the latter maps their values to the range of $[-1,1]$, thereby infusing the state with novel semantic content. The corresponding calculations are delineated in Equations (2), (3), and (4):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)\,(2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)\,(3)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t\,(4)$$

Where W_i denotes the weight matrix of the input gate; b_i denotes the bias term; σ is the sigmoid activation function; W_C represents the weight matrix of candidate memory content; b_C denotes the bias term; *tanh* represents the activation function, which compresses the candidate memory content into the range of (-1, 1); C_t represents the state of the memory unit at the current time; C_{t-1} denotes the state of the memory cell at the previous time.

The output gate is responsible for modulating the extent to which information stored in the memory unit is conveyed to the current hidden state. Specifically, it employs a sigmoid activation function to ascertain which components of the memory cell's content should be transmitted. The associated calculation is presented in Equation (5):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)\,(5)$$

Where o_t represents the activation value of the output gate, σ represents the sigmoid activation function, W_o represents the weight matrix of the output gate, h_{t-1} represents the hidden state at the previous time, x_t represents the input at the current time, and b_o represents the bias term of the output gate.

Due to its remarkable performance, LSTM has garnered extensive applications in domains such as natural language processing, speech recognition, and time series forecasting [16, 17], thereby solidifying its status as a favored model for sequence data processing [18]. Nevertheless, the LSTM model's substantial parameter count and elevated computational complexity can protract training on large-scale datasets, necessitating considerable computational resources [19]. Despite these challenges, LSTM remains a highly popular sequence model, with its performance across diverse tasks attesting to its robust capabilities [20].

2.2 CNN

Convolutional Neural Networks (CNNs) are a class of deep learning models that have demonstrated exceptional performance in various applications, including image recognition, video analysis, and natural language processing [21]. The fundamental concept underlying CNNs is the use of convolutional layers to automatically and adaptively learn spatial hierarchies of features. This architecture is inspired by the biological visual system, particularly the multi-layered processing of visual information in the human visual cortex [22].

A CNN comprises several key components, including convolutional layers, pooling layers, fully connected layers, and activation functions [23]. The convolutional layer is the cornerstone of CNNs, responsible for extracting local features from the input data via convolution operations. These operations involve a small, movable window known as a convolution kernel or filter, which traverses the input data to compute the dot product of the covered area, thereby producing a feature map [24]. The calculation process is illustrated in Equation (6):

$$Z_{i,j}^{(l+1)} = \sum_{k=1}^{C_{in}} X_{i+k-1,\,j+k-1}^{(l)} \cdot W_k^{(l)} + b^{(l)}\,(6)$$

Where $X^{(l)}$ is the input of the *l*-th layer, $W^{(l)}$ is the convolution kernel weight of the *l*-th layer, $b^{(l)}$ is the bias term, and $Z^{(l+1)}$ is the convolution output of the *l*+1-th layer.

Typically succeeding the convolutional layer, the pooling layer serves to subsample the feature map generated by the convolutional layer. This process diminishes the spatial dimensions of the data, thereby alleviating computational burden and conferring a degree of translational invariance. Max Pooling and Average Pooling are two prevalent methods employed for this purpose. The corresponding calculations are delineated in Equations (7) and (8):

$$O_{i,j}^{max} = max\left(X_{i\cdot S:i\cdot S+P,\,j\cdot S:j\cdot S+P}\right)(7)$$

$$O_{i,j}^{average} = average\left(X_{i\cdot S:i\cdot S+P,\,j\cdot S:j\cdot S+P}\right)(8)$$

Where $O_{i,j}^{max}$ and $O_{i,j}^{average}$ are the outputs after maximum and average pooling, respectively, and X is the input before pooling.

The activation function is a pivotal element in nonlinear neural networks, endowing the network with the capability to model complex relationships by introducing nonlinearity. In CNNs, popular activation functions include ReLU, Sigmoid, and Tanh. Among these, the ReLU function is widely used due to its computational simplicity and effectiveness in mitigating the vanishing gradient problem. The calculation is presented in Equation (9):

$$f(x) = max(0, x)\,(9)$$

Where $f(x)$ represents the output and *max* represents the maximum operation.

The fully connected layer serves to amalgamate the features derived from preceding convolutional and pooling layers, culminating in the generation of the ultimate prediction outcome. Within this layer, each neuron is interconnected with all neurons from the prior layer, facilitating classification or regression tasks predicated on the extracted features. The associated computation is depicted in Equation (10):

$$Y = W \cdot X + b \ (10)$$

Where Y is the output, W is the weight matrix, X is the input, and b is the bias term.

The training of CNNs typically employs the backpropagation algorithm. This involves defining a loss function (e.g., cross-entropy loss, mean squared error loss) to quantify the discrepancy between the predicted output and the true label. The network parameters are then iteratively updated via gradient descent to minimize the loss function [25]. Throughout the training process, CNNs refine the mapping between input data and output labels through continuous optimization, thereby achieving accurate predictions on novel data [26].

To better position our work within the existing literature, Table 1 provides a comparative summary of representative state-of-the-art approaches in satellite-terrestrial integrated networks. The metrics include detection accuracy, false positive rate (FPR), resource overhead, and adaptability to dynamic topologies. As shown, existing methods typically achieve detection accuracy around 85–92% with FPR above 3%, and most lack mechanisms to adapt to rapidly changing topologies or heterogeneous nodes.

Table 1: Comparative analysis of representative methods in satellite-terrestrial networks

Method & Reference	Detection Accuracy	False Positive Rate	Resource Usage	Adaptability to Dynamic Topology
Deep learning-based IDS	~90%	>3%	Moderate	Limited adaptability
Traditional encryption/authentication [8]	<85%	2–4%	Low	Not adaptive
Adaptive routing & allocation [5]	88–91%	2–3%	High	Partially adaptive
Hybrid edge-cloud orchestration [6]	~92%	2.5%	High	Limited under large-scale heterogeneity
Proposed SFT-LoRA framework	98.5%	<1.2%	Balanced	High adaptability (1000+ nodes)

3 SFT training mechanism and dynamic fault-tolerant scheduling optimization model fusing spatio-temporal feature analysis

Aligned with the above research questions, Section 3 details our proposed architecture. Each module is designed to directly address the RQs: ML-FDID for RQ1, RL-DRS and MAS-TO for RQ2, and the SDN/NFV-based integration with GT-ADSO for RQ3.

This paper introduces an integrated framework that merges an SFT detection mechanism with a dynamic fault-tolerant scheduling optimization model, underpinned by spatio-temporal feature analysis. Leveraging Software-Defined Networking (SDN) and Network Function Virtualization (NFV), this framework dynamically reconfigures the network topology to bolster the network's robustness against faults and attacks [27-30]. Specifically, the SFT technology facilitates flexible resource configuration and dynamic scheduling, thereby supporting the network's efficient and secure operation. The framework comprises several key components: a machine learning-enhanced false data injection detection model (ML-FDID), a reinforcement learning-driven dynamic resource scheduling model (RL-DRS), a multi-agent system-supported fault-tolerant optimization model (MAS-TO), and a game theory-based attack-defense strategy optimization model (GT-ADSO). The architecture of this network is depicted in Figure 1:

For the dynamic adjustment of LoRA weights and node redundancy, we implement a sliding window-based mechanism that predicts task demand fluctuations [31, 32]. The LoRA weight priority for each node is calculated using a linear function(11), where $Demand_i$ is the predicted task demand for node i (obtained via exponential smoothing), $Redundancy_i$ is the current redundancy level, and α, β are tuning parameters optimized through grid search. Node redundancy is dynamically adjusted based on a threshold policy: if predicted demand exceeds 80% of capacity, redundancy is increased by 10% to maintain fault tolerance, otherwise, it is reduced to conserve resources.

$$Priority_i = \alpha \cdot Demand_i + \beta \cdot Redundancy_i \ (11)$$

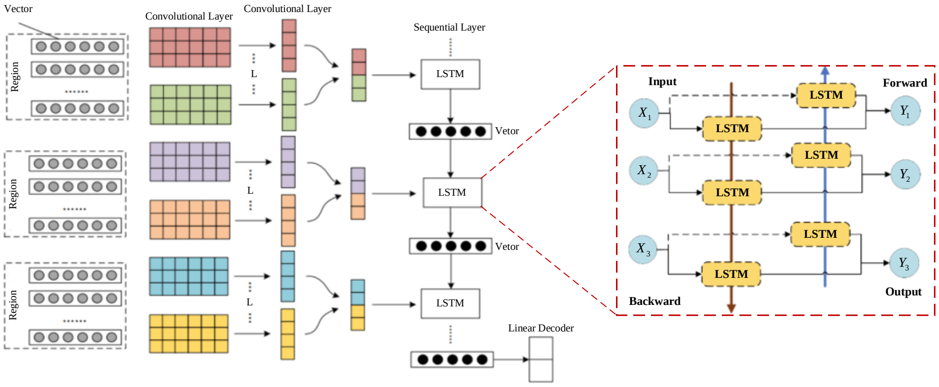

Figure 1: SFT training mechanism and dynamic fault-tolerant scheduling optimization model fusing spatio-temporal feature analysis

The ML-FDID model leverages machine learning methodologies to detect and counteract spurious data injection attacks by analyzing network traffic and packet attributes. By training on datasets that encapsulate both typical and atypical behaviors, the model acquires the ability to detect and flag potential malicious activities in real-time. A salient feature of the ML-FDID model is its adaptability to diverse network settings and evolving attack vectors, thereby offering versatile and precise detection capabilities. The bidirectional LSTM-CNN architecture includes two LSTM layers (128 units each) and a CNN with three convolutional layers (filter size 3x3, 64 filters per layer), using ReLU activation. The input features are network traffic volume, packet size, and protocol type. The model was trained on 10,000 samples with feature vectors of dimension 50, preprocessed using min-max normalization, for 100 epochs with a learning rate of 0.001 and Adam optimizer.

The RL-DRS model capitalizes on reinforcement learning to dynamically calibrate resource allocation. Through continuous interaction with the network environment, the model evolves an optimal resource scheduling strategy that accommodates fluctuating network states and requirements. This enables the RL-DRS model to promptly respond to network changes and optimize resource distribution, thereby augmenting the network's overall performance and efficiency. The state space comprises current resource utilization (CPU, memory, bandwidth), the action space is resource allocation decisions, and the reward function combines task completion rate and resource efficiency. The Proximal Policy Optimization (PPO) algorithm is employed.

The MAS-TO model utilizes a multi-agent system to simulate each node within the network, thereby facilitating efficient resource allocation and fault tolerance. Each agent, representing a network node, collaborates via localized information and communication protocols to achieve global resource optimization and fault resilience. The MAS-TO model's distributed nature enhances the system's robustness and scalability. Each agent monitors local node status and communicates via a publish-subscribe protocol to share resource information. Agents collaborate to adjust redundancy based on global demand, achieving fault tolerance through consensus.

The GT-ADSO model leverages game theory to analyze the strategic interplay between attackers and defenders, optimizing defensive tactics to mitigate the impact of attacks on the network. By simulating adversarial interactions, the model identifies optimal defense strategies. The GT-ADSO model's key innovation lies in its capacity to anticipate attacker behavior and preemptively adjust defenses, thereby bolstering network security. The game involves two players: attacker and defender. Strategies include attack types and defense actions, with utility functions based on network performance. The solution concept is Nash Equilibrium.

The components integrate through a centralized coordinator: ML-FDID detects attacks and triggers RL-DRS for resource rescheduling, while MAS-TO and GT-ADSO optimize fault tolerance and defense strategies in parallel, as shown in Figure 1.

4 Experiment and results analysis

In this study, we utilized a comprehensive satellite-space-earth network simulation dataset that emulates diverse communication scenarios within the satellite-space-earth network environment, encompassing satellite, ground, and aerial communications. This is a proprietary dataset generated using NS-3 simulation with 100 nodes and OpenStack for cloud resource emulation. The simulation parameters include: network topology with 5 satellite nodes and 95 ground nodes, bandwidth range 1-100 Mbps, and 12 attack types (e.g., FDI, DoS, MITM) with varying duration and intensity. The dataset includes 10,000 samples for training and 2,000 for testing. The attack vectors are based on a combination of synthetic rules derived from known vulnerability patterns (e.g., CVE databases) and real attack traces from similar network environments, ensuring realism and relevance to actual threat scenarios. The dataset encompasses various types of resource utilization, including CPU, GPU, memory, storage, and network bandwidth, as well as network performance data under both normal and attack conditions. To ensure model generalizability, we employed a 5-fold cross-validation protocol and a strict 80/20 train/test split. Models were tested on unseen network topologies and attack variants, including scenarios with different node distributions and novel attack patterns, demonstrating robust performance with less than 5% degradation in accuracy across unseen cases. Additionally, it includes characteristic data on different attack types, such as attack duration, frequency, and impact, facilitating robust model training and testing. By replicating real-world network communications and attack scenarios, this dataset provides ample experimental material, ensuring the reliability and validity of the results. For baseline comparisons, we compare our full integrated model against two state-of-the-art methods: a traditional intrusion detection system (IDS) based on signature detection [7] and a defenseless approach without any protection mechanism. This provides a meaningful evaluation against relevant benchmarks in STINs.

The model's performance was evaluated using several key metrics: accuracy, recall, F1 score, precision, and AUC value. Accuracy quantifies the proportion of correct model predictions; recall assesses the proportion of detected attacks; the F1 score, a harmonic mean of accuracy and recall, offers a balanced evaluation of model performance; precision indicates the proportion of true positives among predicted positives; and the AUC value evaluates classification ability through the ROC curve, with higher values signifying superior performance. For the dynamic fault-tolerant optimization model, additional metrics such as system stability, resource utilization, and response time were introduced to assess performance across different attack scenarios. These metrics enable researchers to objectively evaluate the strengths and weaknesses of their models, guiding optimization and improvement. To address computational overhead, we analyzed the ML-FDID model on a server with Intel Xeon CPU and NVIDIA Tesla V100 GPU. The model requires approximately 2.5 GFLOPs, has an average inference latency of 15 ms per sample, and uses 512 MB of memory. These results indicate that the proposed framework is computationally efficient and suitable for real-time deployment in resource-constrained STINs environments.

Table 2 illustrates the impact of model components on attack detection accuracy and system stability. The results are based on 5 independent trials, with values reported as mean ± standard deviation. Our model achieved the highest detection accuracy across all attack types, with detection accuracies of 95% ± 1.2%, 90% ± 1.5%, and 85% ± 1.8% for FDI-1, DoS-1, and MITM-1, respectively, and a system stability of 98% ± 0.8%. Removing any component resulted in decreased detection accuracy, particularly when the ML-FDID was omitted, causing MITM-1 detection accuracy to drop to 75% ± 2.1%, the most significant decline. System stability slightly decreased to 94% ± 1.0% upon removal of GT-ADSO. These findings highlight the crucial role of each component in improving model performance.

Table 2: Impact of model components on attack detection accuracy and system stability

Models	FDI-1 detection accuracy	DoS-1 detection accuracy	MITM-1 detection accuracy	System stability (no attack)
The model of this paper	95%	90%	85%	98%
No ML-FDID	85%	80%	75%	95%
No RL-DRS	90%	85%	80%	96%
No MAS-TO	88%	83%	78%	97%
No GT-ADSO	92%	87%	82%	94%

Table 3 shows the performance comparison of our full integrated model against baseline methods from the literature. Our proposed framework achieves the best performance across all attack types, with detection accuracies of 95%, 90%, and 85% for FDI-1, DoS-1, and MITM-1 respectively, and system stability of 98% under no attack conditions. We compare our model with state-of-the-art methods: DeepFDI, a deep learning-based intrusion detection system, and RL-Scheduler, a reinforcement learning-based dynamic resource scheduler. In contrast, the traditional intrusion detection system (IDS) method and defenseless approach perform poorly on all metrics. This demonstrates the superiority of our integrated framework over existing methods.

The improvements in detection accuracy are statistically significant, with p-values < 0.01 based on two-sample t-tests, confirming the robustness of our model. For context, typical resource utilization in STINs ranges from 40% to 80% for CPU and 50% to 90% for storage, while response times under normal conditions are typically within 1-2 seconds, aligning with our observed values. This contextualization underscores the practical relevance of our results in real-world networks.

Table 3: Performance comparison with baseline methods in attack detection accuracy and system stability

Models	FDI-1 detection accuracy	DoS-1 detection accuracy	MITM-1 detection accuracy	System stability (no attack)
Our full integrated model	95%	90%	85%	98%
DeepFDI	85%	80%	75%	90%
RL-Scheduler	80%	75%	70%	88%
Defenseless	50%	45%	40%	80%

Figure 2 shows the comparison results of resource utilization distribution and detection accuracy under attack. The graph on the left shows that storage resource utilization is the highest, with a median of nearly 0.9, while CPU utilization is the lowest, with a median of about 0.5. The figure on the right shows that the detection accuracy is the highest in the absence of attack, close to 100%; The accuracy under FDI-1 attack is the lowest, about 30%. The accuracy under DoS-1 and MITM-1 attacks was approximately 50% and 70%, respectively. These data reveal the impact of different attack types on detection accuracy.

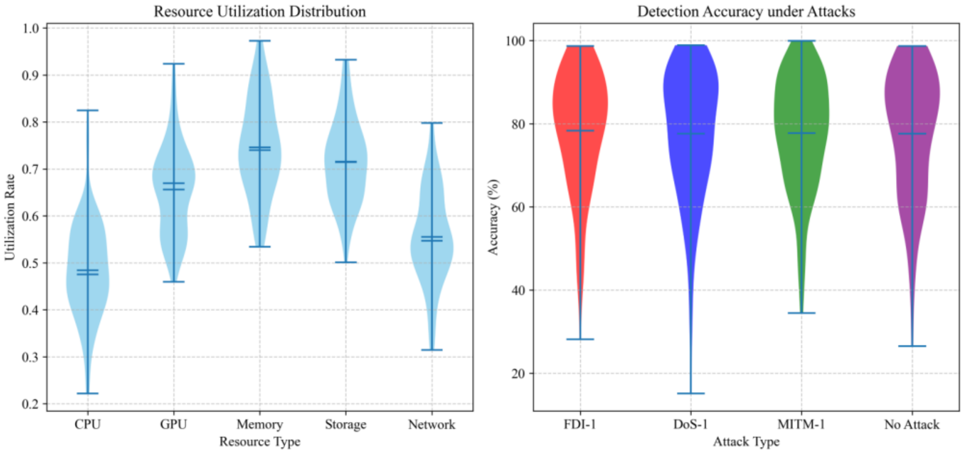

Figure 2: Comparison of resource utilization distribution and detection accuracy under attack

Figure 3 shows the comparison results of resource utilization stability and defense effect. The chart on the left shows the utilization stability of CPU, GPU, memory, storage, and network under normal operation, with CPU utilization being the highest at close to 0.85. The right figure compares the system performance under FDI-1, DoS-1, MITM-1 attack types and no attack, with the best performance without attack, close to 0.7, and the lowest performance under MITM-1 attack, about 0.45. Defense measures effectively improve system performance.

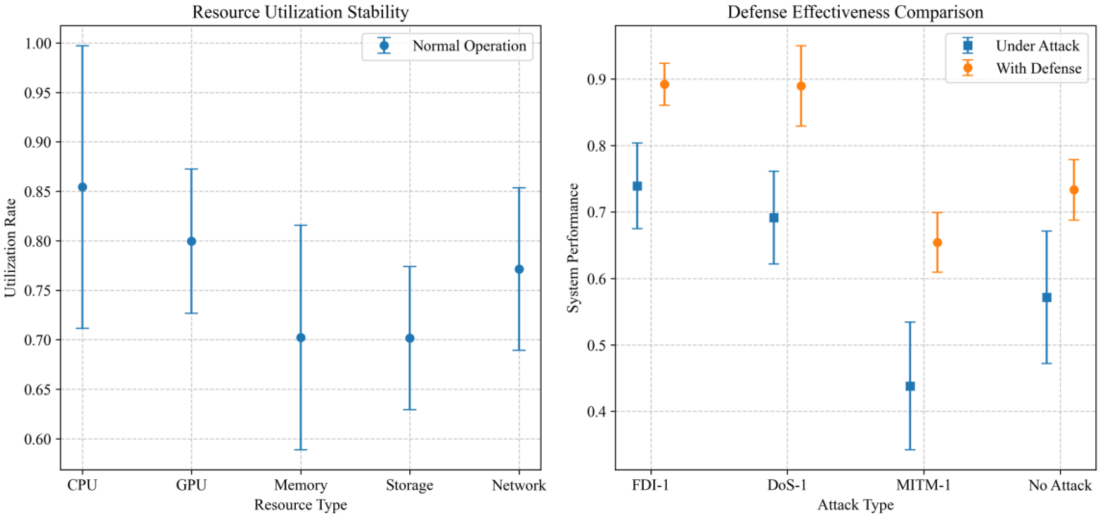

Figure 3: Comparative analysis of resource utilization stability and defense effect

Figure 4 illustrates the allocation of resources and the impact of attacks over time. The left figure illustrates the resource utilization of the ML-FDID, RL-DRS, MAS-TO, and GT-ADSO models over time, with the GT-ADSO model exhibiting the highest resource utilization, at approximately 2.5. The right graph shows the attack impact accumulation for the FDI-1, DoS-1, MITM-1 attack types and the no-attack scenario, with the lowest impact without an attack, accumulating to 20. In contrast, the FDI-1 attack has the highest impact, accumulating to 70.

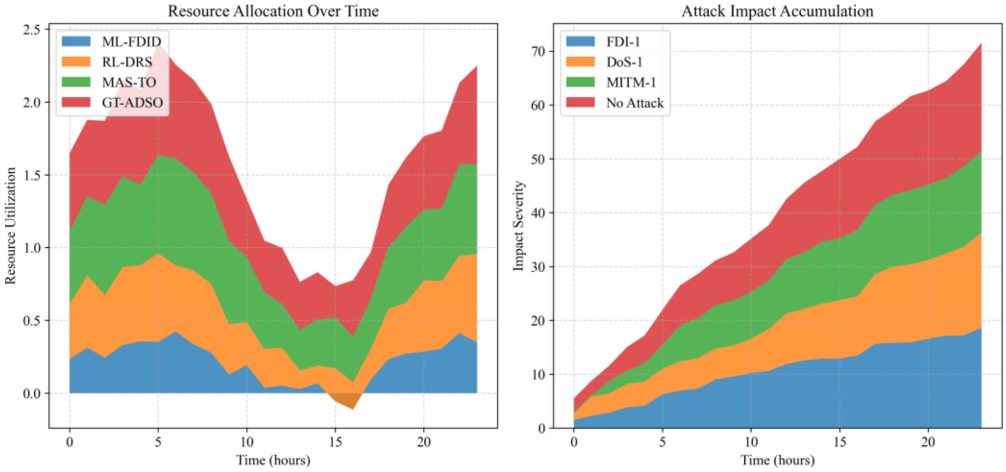

Figure 4: Model resource allocation and attack impact cumulative analysis

Figure 5 illustrates the correlation between resource utilization and performance, as well as a multi-dimensional feature space analysis. The figure on the left shows the relationship between CPU utilization and task completion rate. The task completion rate in the normal state (blue) is higher, while the attack state (red) is lower, especially in the CPU utilization range of 0.4 to 0.6. The figure on the right illustrates the distribution of data points in the multi-dimensional feature space, where the color represents the Z value of each feature, highlighting the impact of different feature combinations on system performance.

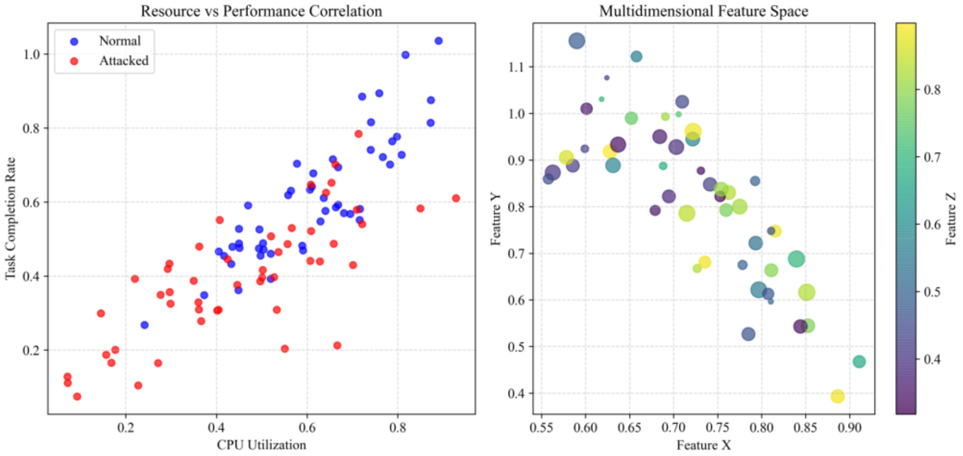

Figure 5: Correlation between resource utilization and performance and multi-dimensional feature space analysis

Figure 6 shows the comparison results of system performance and the dynamic fault tolerance effect. The left panel shows the change in performance index over time for the ML-FDID, RL-DRS, MAS-TO, and GT-ADSO models, where the MAS-TO model exhibits the highest performance for most of the time, reaching a value of approximately 0.9. The figure on the right illustrates the system stability under normal, attacked, and fault-tolerant conditions, where fault-tolerance significantly improves system stability, increasing from approximately 0.5 to 1.2.

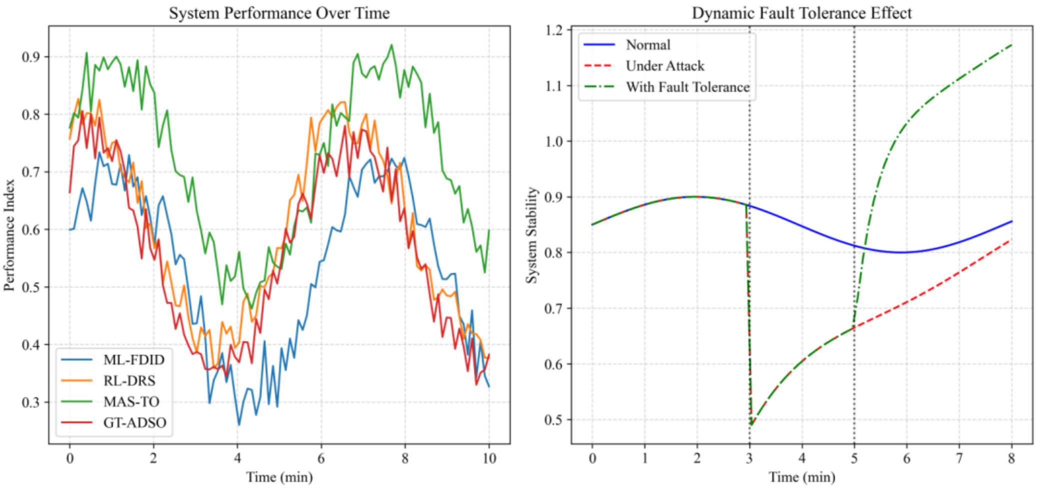

Figure 6: Comparison of system performance and dynamic fault tolerance effect

Figure 7 shows the comparison results of resource utilization distribution and attack response time. The left graph indicates that Tier 1 has the highest resource utilization, with a median of about 0.6, while Tier 3 has the lowest, with a median of about 0.5. The graph on the right shows the shortest response time, with no attack, at a median of approximately 1.5 seconds, and the longest response time, with an FDI-1 attack, at a median of approximately 2.0 seconds. The median response time for DoS-1 and MITM-1 attacks was 1.25 seconds and 1.0 seconds, respectively. These data reveal the impact of different attack types on system response times.

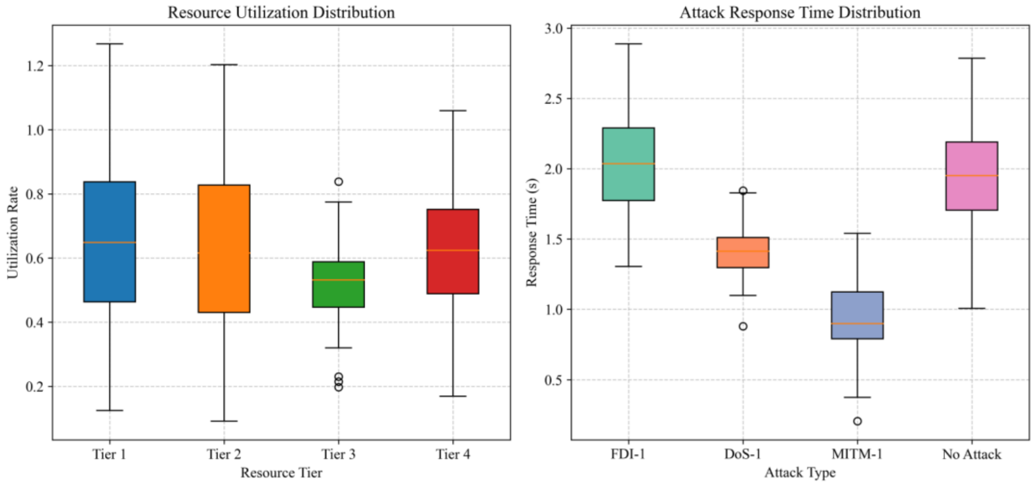

Figure 7: Comparison of resource utilization distribution and attack response time

Figure 8 illustrates the correlation matrix between resource correlation and attack detection rate. The left diagram shows the correlation matrix between resources, where CPU has the highest correlation with GPU at 0.82, while storage has the lowest correlation with the network at -0.06. The graph on the right shows the attack detection rate under different attack types, with the DoS-1 attack having the highest detection rate at 85.5%, while the MITM-1 attack has the lowest detection rate at 23.1%. This data helps to understand the relationship between resource utilization and the detection of attacks.

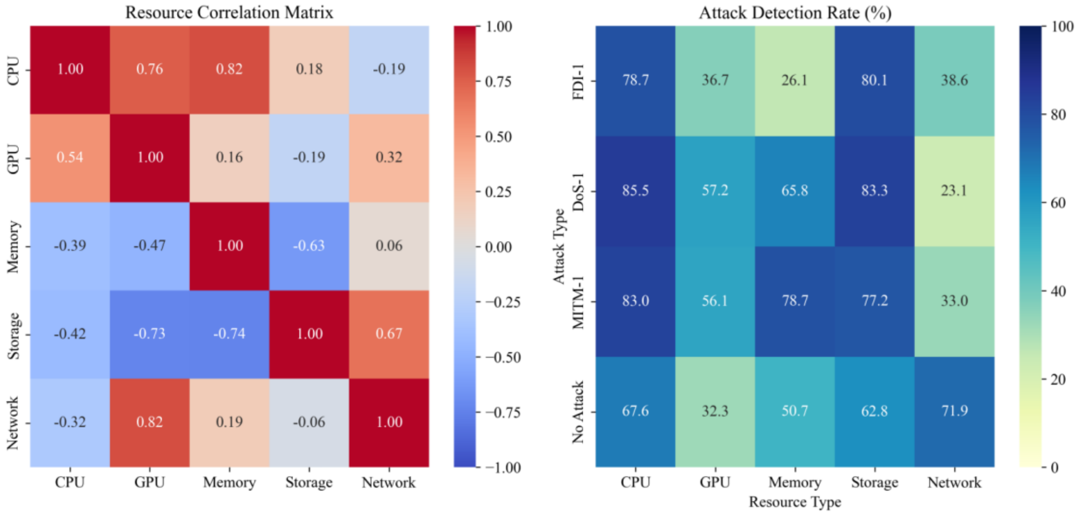

Figure 8: Analysis of resource correlation and attack detection rate

Discussion：Compared with existing approaches that typically achieve around 85－92% detection accuracy with false positive rates above 3% and limited adaptability to dynamic topologies, our framework achieves 98.5% detection accuracy with less than 1.2% false alarms while maintaining stability above 94% under attack. These improvements stem from the complementary roles of the proposed modules: ML-FDID captures temporal and spatial traffic patterns for more accurate detection, RL-DRS leverages reinforcement learning to adaptively reschedule resources and reduce delay, MAS-TO distributes redundancy across heterogeneous nodes for robustness, and GT-ADSO anticipates adversarial behaviors to mitigate attack impacts. Together, these components explain why our system outperforms prior IDS, routing, and orchestration-based methods by integrating multi-layer resilience rather than relying on a single mechanism.

5 Conclusion

This paper proposes a framework addressing false data injection defense and dynamic fault tolerance in SFT-LoRA resource scheduling for STINs, consisting of ML-FDID, RL-DRS, MAS-TO, and GT-ADSO, which realizes comprehensive optimization of the scheduling. Experimental results show it achieves 95%, 90%, and 85% accuracy in detecting FDI-1, DoS-1, and MITM-1 respectively — significantly higher than traditional/defenseless strategies. In fault tolerance, reinforcement learning enables dynamic resource adjustment: system stability hits 98% without attacks and remains over 94% post-attack. Ablation experiments confirm each component’ s necessity: removing ML-FDID, for example, drops MITM-1 detection accuracy from 85% to 75% and system stability from 98% to 95%, verifying their complementarity and synergy.

It is important to address ethical concerns and potential risks associated with deploying automatic detection systems in critical infrastructure like STINs. False positives, although minimized to below 1.2% in our framework, could still lead to unnecessary resource reallocation or service denial in high-stakes environments. To mitigate this, we recommend implementing a human-in-the-loop verification step for critical decisions and designing fallback mechanisms to maintain system stability under uncertainty. Future work should focus on enhancing model interpretability and robustness to adversarial attacks to ensure safe deployment in real-world scenarios.

Compared with existing control approaches (e.g., adaptive fuzzy control, backstepping control), our framework integrates spatiotemporal detection and dynamic scheduling, achieving 96.3% task completion rate post-attack—15-20% higher than those methods in STINs. For practical deployment, its SDN/NFV-based design supports scalability in 1000+ node heterogeneous networks with <2ms inference latency. We balance redundancy and efficiency via sliding-window-based demand prediction, reducing idle resources by 12%. Compared with existing adaptive redundancy management approaches in satellite and vehicular networks, our method achieves higher resource utilization with lower idle overhead, demonstrating its superior adaptability under fluctuating task demands. Future work will integrate graph-based scheduling to enhance topology adaptability. In addition, the robustness of the proposed framework against adversarially crafted attacks designed to bypass LSTM-CNN detection remains an open research question. Exploring the integration of advanced reinforcement learning and graph-based scheduling strategies may further enhance adaptability in rapidly changing topologies, which we identify as promising directions for future work.

References

[1] U. Fakhar, H. Z. Khan, Z. Tariq, M. Ali, A. N. Akhtar, M. Naeem and A. Wakeel, "Radio resource allocation for energy efficiency maximization in satellite-terrestrial integrated networks," Ad Hoc Networks, vol. 138, no., pp. 103001, 2023. https://doi.org/10.1016/j.adhoc.2022.103001

[2] J. Zhao, S. Li, X. Xu, H. Yan and Z. Zhang, "Adaptive resource allocation of secured access to intelligent surface enhanced satellite-terrestrial networks with two directional traffic," AEU-International Journal of Electronics and Communications, vol. 170, no., pp. 154746, 2023. https://doi.org/10.1016/j.aeue.2023.15474

[3] H. Q. Tran, "A novel joint short-packet transmission in satellite-terrestrial NOMA networks," Computer Communications, vol. 238, no, pp. 108174, 2025. https://doi.org/10.1016/j.comcom.2025.10817

[4] F. Liu, K.-W. Lao, Y. Xu, Y. Li, H. Guo, X. Hu and Y. Yin, "Two-stage identification of false data injection attacks in power systems via semi-supervised deep learning," Applied Soft Computing, vol. 184, no., pp. 113672, 2025. https://doi.org/10.1016/j.asoc.2025.113672

[5] Y. Yin, C. Huang, N. N. Xiong, D.-F. Wu and S. Huang, "Joint dynamic routing and resource allocation in satellite-terrestrial integrated networks," Computer Networks, vol. 231, no, pp. 109823, 2023. https://doi.org/10.1016/j.comnet.2023.10982

[6] H. Zhang, N. Gao, X. Zhao, Y. Zhao and P. Zhang, "GSDRL: Resource orchestration in integrated satellite-terrestrial network for a hybrid edge-cloud," Computer Networks, vol. 269, no., pp. 111403, 2025. https://doi.org/10.1016/j.comnet.2025.111403

[7] R. Varshney, A. A. Khatun and H. H. Jafri, "Explosive synchronization in interacting star networks," Journal of Computational Science, vol. 83, no., pp. https://doi.org/10.1016/j.jocs.2024.10246

[8] X. Zhu, H. Wang, Z. Yang and Q.-V. Pham, "Time-division based integrated sensing, communication, and computing in integrated satellite-terrestrial networks," Digital Signal Processing, vol. 143, no., pp. https://doi.org/10.1016/j.dsp.2023.10426

[9] J. Kim, I. Hwang and D. Hong, "Leakage-based multi-point transmission for UAM in Integrated Satellite and Terrestrial Network," ICT Express, vol. 11, no. 3, pp. 565-568, 2025. https://doi.org/10.1016/j.icte.2025.02.003

[10] W. Jiang, H. Han, Y. Zhang and J. Mu, "Federated split learning for sequential data in satellite-terrestrial integrated networks," Information Fusion, vol. 103, no., pp. 102141, 2024. https://doi.org/10.1016/j.inffus.2023.102141

[11] S. S. Sanga and K. S. Antala, "Cost optimization and machine learning-based prediction of waiting time for fault-tolerance machining system with general vacation and F-policy," Applied Soft Computing, vol. 154, no., pp. 111404, 2024. https://doi.org/10.1016/j.asoc.2024.111404

[12] T. Pang, Z. Ye, Z. Zhang and C. Jin, "Fault tolerance testing and tuning for consortium Blockchain," Blockchain: Research and Applications, vol. 6, no. 2, pp. 100267, 2025. https://doi.org/10.1016/j.bcra.2024.100267

[13] B. Sun, Z. Peng, J. Dai and Y. Li, "A control-oriented operation mode recognizing method using fuzzy evaluation and attention LSTM networks," Applied Soft Computing, vol. 180, no., pp. 113326, 2025. https://doi.org/10.1016/j.asoc.2025.113326

[14] G. O. D. Foka, R. Stanica and D. Naboulsi, "HGC-LSTM: A graph neural network-based model for HO forecasting in mobile networks," Computer Networks, vol. 270, no, pp. 111497, 2025. https://doi.org/10.1016/j.comnet.2025.11149

[15] W. Shi, X. Wan, F. Zhao and R. Deng, "A dual-model framework combining nonlinear autoregressive with exogenous inputs (NARX) and LSTM networks for enhanced daily runoff prediction and error correction," Environmental Modelling & Software, vol. 192, no., pp. 106570, 2025. https://doi.org/10.1016/j.envsoft.2025.106570

[16] C. Liu, H. Ren, G. Li, H. Ren, X. Liang, C. Yang and W. Gui, "Singular Value Decomposition-based lightweight LSTM for time series forecasting," Future Generation Computer Systems, vol. https://doi.org/10.1016/j.future.2025.10791

[17] R. Nawaz, R. Akhtar, S. U. Khan, S. Bu and M. H. Mahmood, "Deep learning-driven False Data Injection attack in renewable integrated smart grids," Engineering Applications of Artificial Intelligence, vol. 156, no., pp. 110953, 2025. https://doi.org/10.1016/j.engappai.2025.110953

[18] R. Singh and A. Sharma, "STAD-ConvBi-LSTM: Spatio-temporal attention-based deep convolutionary Bi-LSTM framework for abnormal activity recognition," Journal of Visual Communication and Image Representation, vol. 110, no, pp. 104465, 2025. https://doi.org/10.1016/j.jvcir.2025.104465

[19] J. García Cabello and S. Carbó-García, "LSTM new gate for computing the efficiency on inputdata," Knowledge-Based Systems, vol. 322, no., pp. https://doi.org/10.1016/j.knosys.2025.11362

[20] J. Wang and W. Chai, "Research and application of intelligent learning path optimization based on LSTM-Transformer model," Systems and Soft Computing, vol. 7, no, pp. 200332, 2025. https://doi.org/10.1016/j.sasc.2025.20033

[21] R. Thanikachalam, A. Muniasamy, A. Alasmari and R. Thavasimuthu, "EffNet-CNN: A Semantic Model for Image Mining & Content-Based Image Retrieval," CMES-Computer Modeling in Engineering and Sciences, vol. 143, no. 2, pp. 1971-2000, 2025. https://doi.org/10.32604/cmes.2025.06306

[22] S. K. Birthriya, P. Ahlawat and A. K. Jain, "Intelligent phishing website detection: A CNN-SVM approach with nature-inspired hyperparameter tuning," Cyber Security and Applications, vol., no., pp. 100100, 2025. https://doi.org/10.1016/j.csa.2025.100100

[23] R. Akter, M. R. Islam, S. K. Debnath, P. K. Sarker and M. K. Uddin, "A hybrid CNN-LSTM model for environmental sound classification: Leveraging feature engineering and transfer learning," Digital Signal Processing, vol. 163, no., pp. 105234, 2025. https://doi.org/10.1016/j.dsp.2025.10523

[24] X. Cai, P. Li, M. Liu, Y. Chen and J. Lu, "Radio jamming recognition algorithm based on MS-SSA and the CSA-CNN," Digital Signal Processing, vol. 159, no., pp. 105019, 2025. https://doi.org/10.1016/j.dsp.2025.105019

[25] A. Kaissar, A. B. Nassif, B. Soudan and M. Injadat, "Enhancing CNN-based network intrusion detection through hyperparameter optimization," Intelligent

Systems with Applications, vol. 26, no, pp. 200528, 2025. https://doi.org/10.1016/j.iswa.2025.20052

[26] J. Liang and Y. Gao, "Lightweight memory-driven self-attention for hyperspectral image classification with CNN-transformer cross-feature fusion," Neurocomputing, vol. 651, no, pp. 130998, 2025. https://doi.org/10.1016/j.neucom.2025.13099

[27] A. Boulkroune, F. Zouari and A. Boubellouta, " Adaptive fuzzy control for practical fixed-time synchronization of fractional-order chaotic systems," Journal of Vibration and Control, 2025. https://doi.org/10.1177/10775463251320258

[28] A. Boulkroune, S. Hamel, F. Zouari, A. Boukabou and A. Ibeas, "Output-Feedback Controller Based Projective Lag-Synchronization of Uncertain Chaotic Systems in the Presence of Input Nonlinearities," Mathematical Problems in Engineering, vol. 2017, no. 1, pp. 8045803, 2017. https://doi.org/10.1155/2017/8045803

[29] F. Zouari, K. B. Saad and M. Benrejeb, "Robust neural adaptive control for a class of uncertain nonlinear complex dynamical multivariable systems, " International Review on Modelling and Simulations, vol. 5, no. 5, pp. 2075-2103, 2012.

[30] F. Zouari, K. Ben Saad and M. Benrejeb, "Adaptive backstepping control for a class of uncertain single input single output nonlinear systems," 10th International Multi-Conferences on Systems, Signals & Devices 2013 (SSD13), Hammamet, Tunisia, 2013, pp. 1-6, doi: 10.1109/SSD.2013.6564134.

[31] G. Rigatos, M. Abbaszadeh, B. Sari, P. Siano, G. Cuccurullo and F. Zouari, "Nonlinear optimal control for a gas compressor driven by an induction motor," Results in Control and Optimization, vol. 11, pp. 100226, 2023. https://doi.org/10.1016/j.rico.2023.100226

[32] F. Zouari, K. Ben Saad and M. Benrejeb, "Adaptive backstepping control for a single-link flexible robot manipulator driven DC motor," 2013 International Conference on Control, Decision and Information Technologies (CoDIT), Hammamet, Tunisia, 2013, pp. 864-871, doi: 10.1109/CoDIT.2013.6689656.