computers

Article

A Lightweight Intrusion Detection System for IoT and UAV Using Deep Neural Networks with Knowledge Distillation

Treepop Wisanwanichthan and Mason Thammawichai *

Electrical Engineering Department, Navaminda Kasatriyadhiraj Royal Air Force Academy, Saraburi 18180, Thailand; treepop@rtaf.mi.th

* Correspondence: mason@rtaf.mi.th

Abstract

Deep neural networks (DNNs) are highly effective for intrusion detection systems (IDS) due to their ability to learn complex patterns and detect potential anomalies within the systems. However, their high resource consumption requirements including memory and computation make them difficult to deploy on low-powered platforms. This study explores the possibility of using knowledge distillation (KD) to reduce constraints such as power and hardware consumption and improve real-time inference speed but maintain high detection accuracy in IDS across all attack types. The technique utilizes the transfer of knowledge from DNNs (teacher) models to more lightweight shallow neural network (student) models. KD has been proven to achieve significant parameter reduction (92–95%) and faster inference speed (7–11%) while improving overall detection performance (up to 6.12%). Experimental results on datasets such as NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, and UAV IDS demonstrate DNN with KD's effectiveness in achieving high accuracy, precision, F1 score, and area under the curve (AUC) metrics. These findings confirm KD's ability as a potential edge computing strategy for IoT and UAV devices, which are suitable for resource-constrained environments and lead to real-time anomaly detection for next-generation distributed systems.

Keywords: anomaly detection; deep neural network; information security; internet of things; intrusion detection system; knowledge distillation; network security; lightweight model; unmanned aerial vehicle

check for updates

Academic Editor: Paolo Bellavista

Received: 15 June 2025
Revised: 15 July 2025
Accepted: 15 July 2025
Published: 19 July 2025

Citation: Wisanwanichthan, T.; Thammawichai, M. A Lightweight Intrusion Detection System for IoT and UAV Using Deep Neural Networks with Knowledge Distillation. *Computers* **2025**, *14*, 291. https://doi.org/10.3390/computers14070291

1. Introduction

An intrusion detection system (IDS) is a security tool that helps monitor and analyze the activities of a network or system for malicious actions. This implies that IDS is a vital section of cybersecurity in the way that it can alert administrators when potential threats occur [1]. There are two main types of IDS: signature-based and anomaly-based. Signature-based IDS checks traffic against pre-defined patterns and anomaly-based IDS operates by learning normal network activities and alerts once any deviations are found [2]. An anomaly IDS excels at identifying unknown threats like zero-day attacks that lack exact-matched patterns, including attacks in IoT systems [3]. Therefore, IDS is often seen as another layer of defense to complement other security measures, protecting systems against serious cyber threats [4,5]. Deep learning utilizes artificial neural networks with multiple layers of neurons to optimize pattern recognition and understand sophisticated data. The system processes a substantial amount of data as well as extracts key characteristics automatically [6]. Several architectures like convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have reached remarkable results in image analysis, and text processing besides spotting anomalies [7]. Moreover, deep learning is highly suitable for anomaly-based IDS since it can learn complex patterns of normal and abnormal behaviors from high-dimensional data. Unlike traditional approaches, which rely on predefined rules or signatures, deep learning can adapt to unseen threats by identifying deviations from learned patterns. This makes it highly effective in detecting potential zero-day attacks and other novel threats based on the probability output from supervised neural network models [8]. Though there are certain challenges like high computational cost and the risk of overfitting, the advantages of deep learning in achieving high detection accuracy and its adaptability can be one of the best choices for anomaly-based IDS [9].

Model compression refers to techniques that reduce the size of a deep learning model while maintaining its performance. This is important for model usage in resource-constrained settings, e.g., in a mobile device or IoT system. Often, the practices include pruning, quantization, and knowledge distillation. Knowledge distillation (KD) involves knowledge transferring between a large and complex model (teacher) into a smaller, simpler model (student). In the process, the student model also learns from the output labels and probability distributions of the teacher model over classes that are used as indication of attacks [10]. Adopting KD in edge computing is particularly important due to limited computing and energy resources. By using lightweight models instead of large ones, inference can occur locally, reducing reliance on high-latency cloud computing. This can lead to faster response times, enhanced data privacy, and reduced bandwidth usage, which are critical for applications like real-time anomaly detection in IoT networks [11]. Furthermore, smaller models consume less power, satisfying energy efficiency requirements of edge devices. Its ability to balance model performance with computational efficiency makes it a promising approach for future advancements in edge computing [12,13]. Deep neural network (DNN) models are extremely computationally expensive, require heavy memory, and are slow during inference, so they are not suitable for real-time applications running on low-power devices like IoT sensors or UAVs. Large-scale model deployment is also highly problematic in an energy-constrained environment because large models consume extensive energy both at training and inference [14]. While shallow neural networks are lighter and faster, they often under fit complex, high dimensional data and therefore may fail to match the generalization performance of deeper models, which have greater representational capacity. These limitations highlight the need for techniques such as knowledge distillation. Additionally, smaller models require less energy, which aligns well with the goals of energy efficiency and sustainability [15].

Knowledge distillation (KD) can be a promising solution to address both scalability and performance limitations of DNN models, making it especially relevant for deploying intelligent systems in resource-constrained environments such as IoT and UAV platforms [16]. These environments demand real-time processing and low computational overhead that traditional DNNs often fail to meet due to their high complexity and inference latency. To address these issues, this paper presents the implementation of a KD-based framework for developing lightweight yet effective intrusion detection systems tailored to IoT and UAV settings. Specifically, we (1) evaluate the effectiveness of KD in enabling compact DNN models to perform real-time threat detection with minimal resource usage, (2) demonstrate the framework's generalizability by applying it across multiple benchmark intrusion detection datasets, including NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, and UAV IDS, (3) show that the distilled student models maintain high detection accuracy while reducing model size by over 93%, making them suitable for edge deployment, and (4) enhance the models' capability to detect rare and potentially unseen attacks by leveraging soft targets from teacher models, addressing a critical need in security for low-power systems.

The paper is structured as follows: Section 2 reviews related works on anomaly-based IDS using neural networks and KD. Section 3 explains the methodology, including datasets, data preprocessing, and the KD algorithm. Section 4 presents and analyzes the results across five datasets: NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, and UAV IDS. Finally, the conclusion is provided in Section 5.

2. Related Work

Recent advancements in intrusion detection have produced a wide variety of techniques aimed at balancing detection accuracy with computational efficiency, particularly for resource-constrained environments like IoT and UAV systems. This section organizes existing work into five categories based on the main techniques employed, i.e., knowledge distillation (KD), federated learning with KD, hybrid and GAN-based methods, cyber-physical feature fusion, and architecture-specific or optimization-driven distillation. This categorization highlights the strengths, limitations, and trends across current approaches. All references in each category are sorted by year (earliest first).

2.1. Knowledge Distillation-Based IDS

A growing number of studies apply knowledge distillation to compress deep models while maintaining competitive detection performance. A KD-based lightweight IDS model was designed by Zhao et al. [17] by using a separation convolution technique for reducing model size and computation cost by 99% and has achieved state-of-the-art accuracy of 94.3% on the UNSWNB15 and 91.46% on KDDCUP99 dataset. Wang et al. [14] proposed a KD-based triplet convolutional neural network for industrial CPS, which can reduce the model size by 86% with only 0.4% loss of accuracy and showed better performance than state-of-the-art methods on the NSL-KDD and CIC IDS2017 datasets. Chen et al. [18] proposed a semi-supervised KD framework in campus networks. The work realized efficient intrusion detection based on the distributed deployment at the aggregation layer and leveraged labeled and unlabeled data in training. The framework resulted in 98.49% parameter reduction and 98.52% decrease in FLOPs for the student models compared to the teacher models while only slightly decreasing in accuracy of around 1–2%. Zhu et al. [19] proposed LKD-STNN, a lightweight KD-based model for malicious traffic detection, which was able to obtain accuracy above 98% on the ToN-IoT and IoT-23 datasets while reducing the model parameters to less than 1% from the teacher model, thus proving ideal for deployment at the edge devices. Ren et al. [20] propose a KD-based framework to detect GPS spoofing attacks in small UAVs using a compact neural network suitable for resource-constrained platforms. A long short-term memory (LSTM) model is first trained on telemetry data (GPS, IMU, RSSI) and then distilled into a lightweight neural network (student) using KD. Experimental results on the UAV IDS dataset from IEEE DataPort show that the student model trained with LSTM achieves 95.74% reduction in model size (from 72.7 KB to 3.0 KB) and 31.63% reduction in memory overhead. Wang et al. [16] proposed a CNN-based model with two-dimensional Fourier transform and KD, which has less computational cost while maintaining competitive accuracy in several public datasets. The teacher model consisted of an eight-layer CNN compared to a one-layer CNN in the student model. The student model can reduce the number of parameters from around 1.7 million to only one thousand, while maintaining overall accuracy in every tested dataset. Abbasi et al. [21] demonstrated the effectiveness of KD in IoT traffic classification, where student models trained using KD achieved similar accuracy to larger teacher models from 93.57% to 92.45% but with significant savings in computational cost and storage requirements.

2.2. Federated Learning with Knowledge Distillation

To support decentralized and privacy-aware deployment, federated learning (FL) has been combined with KD. Zhao et al. [22] proposed a semi-supervised federated learning (SSFL) scheme with KD. This achieved a precision of 91.73% on IoT traffic datasets compared to that of traditional methods in federated learning such as MLP with only 87.57%. Shen et al. [23] proposed an ensemble KD-based federated learning framework called FLEKD, which achieved 99.8% accuracy in CIC-IDS2019 and further improved the detection of the most difficult attacks, i.e., UDPLag by 80.86% while ensuring challenges such as data heterogeneity in IoT networks. Quyen et al. [24] developed the robust federated learning framework of FedKD-IDS, introducing KD and anti-poisoning mechanisms to overcome some of the challenges faced with non-independent data and poisoning attacks. They achieved an accuracy of 79% in cases of poisoning attacks on the N-BaIoT dataset and outperformed traditional semi-supervised federated methods by a large margin.

2.3. Hybrid and GAN-Based IDS with KD Approaches

Some works integrate KD with adversarial learning and generative models. Ravipati and Abualkibash [25] proposed KDDT, which is a knowledge distillation-empowered digital twin model in anomaly detection problems in train control systems. By combining KD with a pre-trained variational autoencoder, the system achieved F1 scores of 93.1% and 91.5% on two datasets, demonstrating its effectiveness in extracting both contextual and temporal features. Ali et al. [26] proposed a lightweight intrusion detection system that combined a generative adversarial network (GAN) and KD to handle adversarial threats in IoT and IIoT networks. Their lightweight method maintained high accuracy, above 95% in the classic NSL-KDD and CIC-IDS2017, while ensuring resource efficiency across datasets like CIC-IDS2017 and IoT-23.

2.4. Cyber-Physical and Feature Fusion Models

Another stream of research focuses on combining cyber and physical domain features for improved intrusion detection. Bertoli et al. [27] introduced AB-TRAP, an end-to-end framework for ML-based NIDS that systematically covers key aspects such as realization on target devices (kernel-space Linux modules or user-space NFQUEUE) and performance evaluation on both LAN and Internet scenarios. They demonstrate deployment on a Raspberry Pi in achieving F1-scores up to 0.96, AUC = 0.98, and only 1–4% CPU/RAM overhead highlighting practical considerations for real-world, resource-constrained intrusion detection system. Hassler et al. [28] developed an intrusion detection system that fuses both cyber and physical features to enhance the security of UAVs. The authors created a new cyber-physical dataset capturing UAV operations under normal and attack conditions. They evaluated various machine learning models and found that integrating cyber and physical data significantly improves detection performance, especially against unseen attacks of varying complexity. The results underscore the effectiveness of feature fusion in securing cyber-physical systems. Hadi et al. [29] introduced autonomous collaborative IDS (UAV-CIDS), a real-time collaborative IDS using a feedforward CNN to detect attacks on UAV networks. UAV-CIDS analyzes encoded Wi-Fi traffic from three commercial drone models (DBPower UDI, Parrot Bebop, DJI Spark) and is trained to spot zero-day threats. On the UAVIDS dataset of real UAV traffic, the FFCNN model attains 98.23% accuracy. He noted challenges in detecting UAV zero-day attacks as it lacks features like next-generation networks (NGNs, 5G/6G).

2.5. Self-Knowledge Distillation and Optimization-Based Distillation

Several works emphasize self-distillation and architectural optimization. Yang et al. [30] proposed a lightweight self-knowledge distillation model, LNet-SKD, that accomplished high accuracy while reducing computation overhead. Li and Yao [15] designed a two-stage lightweight intrusion detection model using self-supervised contrastive learning and self-knowledge distillation in IoT networks, which has higher accuracy in intrusion detection and attains better generalization among various public datasets such as KDD CUP99, NSL-KDD, UNSW-NB15, CIC IDS2017, and BoT-IoT. Their methodology could achieve up to 99.95% in accuracy in binary classification on KDD CUP99. Wang el al. [31] proposed the TBCLNN model, which uses self-knowledge distillation to build a lightweight intrusion detection system for IoT networks. The method utilizes the binary Harris hawk optimization algorithm for feature selection and employs tied block convolution to design efficient neural network blocks. An improved self-knowledge distillation loss function is introduced to mitigate sample imbalance issues. The model achieves multiclassification accuracies above 99% on three public IoT datasets while being computationally efficient. These studies underline the growing role of KD in balancing performance and efficiency in IDS, particularly for deployment in IoT and edge computing environments.

Despite significant progress in intrusion detection research, existing IDS models face several key limitations. Traditional machine learning models often rely on manual feature engineering and fail to generalize well to complex threats. Deep learning-based IDSs can be offering high accuracy but typically computationally expensive and unsuitable for deployment in resource-constrained environments such as IoT devices or UAV systems. Ensemble and hybrid models tend to improve detection performance but increase model size and inference time, limiting their real-time applicability. To address these challenges, we conducted a study on a KD-based IDS framework that compresses high-capacity teacher models into lightweight student models while preserving strong detection performance. We further demonstrate the framework's practicality by evaluating it across multiple benchmark intrusion detection datasets.

Our study differs from existing work by focusing on the implementation of our predefined deep neural network architecture integrated with KD and evaluating it across multiple well-known and domain-specific intrusion detection datasets, including NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, and the UAV IDS dataset. Unlike prior studies that typically explore accuracy improvements or use KD with other techniques, our study demonstrates that a standard deep learning (fully-connected neural networks) structure can consistently work with the KD technique to achieve strong detection performance, faster inference speed, and significant parameter reduction. A comparison table of key KD-based IDS studies and our work is shown in Table 1. Also, we are the first to systematically apply a vanilla KD framework across five diverse IDS datasets (traditional networks, IoT, and UAV), all under the same architecture and hyperparameters to demonstrate cross-domain generality without overparameterization.

Table 1. Comparison of key KD-based IDS studies and this work.

Study	Datasets Used	Methods	Key Contributions
Wang et al. (2022) [14]	NSL-KDD, CIC-IDS2017	Triplet-CNN teacher to small CNN student via KD	Reduced model size by 86% with only 0.4% accuracy loss; outperformed SOTA on both benchmarks.

Table 1. *Cont.*

Study	**Datasets Used**	**Methods**	**Key Contributions**
Chen et al. (2023) [18]	Campus network traffic	Semi-supervised KD with labeled and unlabeled data	Realized 98.49% parameter reduction and 98.52% FLOPs decrease for student models, with only 1–2% drop in accuracy.
Li & Yao (2024) [15]	KDD CUP99, NSL-KDD, UNSW-NB15, BoT-IoT, CIC-IDS2017	CL-SKD two-stage detection: contrastive self-supervision, self-KD, and depth-wise separable convolution	Achieved up to 99.95% in binary and 99.93% in multi-class accuracy on KDD CUP99 ,and strong generalization across multiple IDS datasets.
Ali et al. (2024) [26]	NSL-KDD, CIC-IDS2017, IoT-23	Generative adversarial networks with KD	Demonstrated >95% accuracy in NSL-KDD and CIC-IDS2017 datasets and 89% in IoT-23, but performance drops with adversarial training.
Wang et al. (2025) [31]	CIC-IDS2017, BoT-IoT, TON-IoT	Self-KD (TBCLNN) with binary Harris hawk optimization for feature selection and tied block convolution	Obtained 99% multiclass classification accuracy across all three benchmarks while remaining computationally efficient.
This Work	NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, UAV-IDS	Four-layer FCN teacher to two-layer FCN student via KD (alpha = 0.4, T = 3)	First to apply vanilla KD uniformly across five diverse IDS domains, achieving > 90% parameter reduction and 11% faster inference without accuracy loss.

3. Methodology

3.1. Dataset Introduction

3.1.1. NSL-KDD

NSL-KDD is a refined version of the KDD'99 dataset, addressing problems such as redundant records and imbalance. It has been widely used in recent years for the evaluation of IDS, since it contains a balanced distribution of attack types, reducing the chances of biased model training for intrusion detection [32]. These four categories, namely, DoS, Probe, R2L, and U2R, represent most of the common intrusion scenarios. The diversity of the attacks enables IDS models to be trained and tested on a wide range of realistic cyber threats [25].

NSL-KDD is divided into two files: the *KDDTrain+*, containing 125,973 records for training, and the *KDDTest+*, which contains 22,544 records for testing. One of the challenges of this dataset is that some of the attack types in the test data are not represented in the training data. Interestingly, this reflects real-world situations where the IDS will be expected to detect new threats without having to have seen them before. Therefore, models tested on NSL-KDD must generalize and be flexible, and this dataset is a good benchmark for evaluating the performance of IDS in dynamic environments [33]. The attack categories presented in this dataset are shown in Table 2.

Table 2. NSL-KDD [32] class descriptions and data distribution.

Class	**Description**	**No. of Training**	**No. of Testing**
Denial of Service (DoS)	Attacks that make the service unavailable by flooding connections.	45,927	7460
Normal	Normal traffic.	67,343	9711
Probe	Scans or probes for vulnerabilities or important data (e.g., port scanning).	11,656	2421
Remote to Local (R2L)	An outsider gains unauthorized local access.	995	2885
User to Root (U2R)	A local user account is exploited to gain root or administrative privileges.	52	67

3.1.2. UNSW-NB15

The UNSW-NB15 dataset is a modern benchmark for evaluating intrusion detection systems (IDS), created to address the limitations of older datasets. It provides a realistic representation of more modern network traffic and normal behaviors. The dataset includes nine distinct types of attacks—analysis, backdoor, DoS, exploits, fuzzers, generic, Reconnaissance, Shellcode, and worms—offering a broad category of threats for IDS evaluation [34]. It provides diversity that adds strength and adaptability to the IDS models to evolve with cyberattacks, making UNSW-NB15 an appropriate choice for standardized evaluation [35].

There are two files: *UNSW_NB15_training-set.csv*, which contains 175,341 records for training, and *UNSW_NB15_testing-set.csv*, with 82,332 records for testing. This separation simulates real-world conditions where IDS must identify both known and unknown threats in live environments. The comprehensive variety of attack types and the realistic nature of the traffic data ensure that IDS models trained and tested on UNSW-NB15 can generalize effectively [36]. The attack categories presented in this dataset are shown in Table 3.

Table 3. UNSW-NB15 [34] class descriptions and data distribution.

Class	**Description**	**No. of Training**	**No. of Testing**
Analysis	Evaluating system behavior or patterns to identify vulnerabilities or threats.	2000	677
Backdoor	A hidden entry point used to access a system without authorization.	1746	583
DoS	Flooding a system or resource to deny service to legitimate users.	12,264	4089
Exploits	Attacks that take advantage of vulnerabilities in software or systems.	33,393	11,132
Fuzzers	Tools or methods that send random data to applications to uncover weaknesses.	18,184	6062
Generic	Attacks that are not specific to a particular system or application.	40,000	18,871
Normal	Normal traffic.	56,000	37,000
Reconnaissance	Gathering information about a target to prepare for an attack.	10,491	3496
Shellcode	Malicious code used to control or exploit a compromised system.	1133	378
Worms	Self-replicating malware that spreads without requiring user interaction.	130	44

3.1.3. CIC-IDS2017

The CIC-IDS2017 dataset is a comprehensive intrusion detection dataset that captures realistic network traffic, reflecting the complexity of modern cyberattacks. It includes 14 different attack categories, such as bot, DDoS, various DoS attacks (e.g., GoldenEye, Hulk), brute force (FTP and SSH Patator), PortScan, and web attacks (brute force, SQL injection, XSS). Despite its breadth, the dataset suffers from an extremely skewed distribution of attack types, which necessitates reorganization for effective evaluation. To address this, all DoS records were grouped into a single DoS category, FTP and SSH Patator records were combined into brute force, and all web attack records were consolidated. Infiltration and Heartbleed attacks were excluded due to their minimal representation (fewer than 0.00001% of total records, i.e., 36 and 11 records, respectively, out of over 2.8 million samples). This restructuring prevents the skew of attacks, to make the model training and evaluation more balanced [9].

Unlike datasets with predefined training and testing splits, CIC-IDS2017 consolidates all traffic data into a single collection of CSV files. After merging the files in *MachineLearningCSV.zip*, the dataset comprises 2,830,743 records. To maintain the integrity of attack distributions, an 80:20 stratified train/test split is employed during evaluation. This approach is to preserve class ratios and ensure that models are trained and tested on data representative of real-world attack patterns. This makes the CIC-IDS2017 dataset valuable in developing and validating intrusion detection systems due to its scale, diversity, and realism [31]. The attack categories presented in this dataset are shown in Table 4.

Table 4. CIC-IDS2017 [9] class descriptions and data distribution.

Class	Description	No. of Training	No. of Testing
Bot	An attack that performs automated tasks, often used in malicious activities like botnets.	1565	391
BruteForce	An attack that systematically tries all possible combinations of passwords or keys.	11,066	2766
DoS	Overwhelming a system with traffic to make it unavailable.	102,420	25,605
DDoS	A DoS attack launched from multiple devices, often part of a botnet.	201,369	50,343
Normal	Normal traffic.	1,817,056	454,264
PortScan	Scanning a system's ports to find open or vulnerable entry points.	127,043	31,761
Web Attack	Exploiting vulnerabilities in web applications, such as through SQL injection or XSS.	1744	436

3.1.4. IoTID20

The IoTID20 dataset is a specialized intrusion detection dataset designed to reflect security threats targeting IoT environments. It captures realistic network traffic and focuses on four primary attack types: DoS, Mirai, MITM ARP spoofing, and scan. This focus on IoT-specific threats makes IoTID20 especially relevant for developing IDS for securing IoT devices and networks, which are increasingly vulnerable to cyber-attacks [37]. Due to the limited computational power, IoT systems require special IDS models unlike traditional IDS for evaluating the presented cyber threats considering IoT-security needs [38].

IoTID20 does not include predefined training and testing splits. Instead, the data is consolidated into a single file, *IoT Network Intrusion Detection.csv*, containing 625,783 records. For evaluation, a stratified 80:20 train/test split is used to maintain the class distribution, ensuring the model is trained and tested on representative samples of each attack type. This balanced splitting approach enhances the reliability of performance assessments and supports the development of robust IDS solutions capable of detecting diverse characteristics of IoT-focused intrusions including package and flow data [39]. The attack categories presented in this dataset are shown in Table 5.

Table 5. IoTID20 [37] class descriptions and data distribution.

Class	Description	No. of Training	No. of Testing
DoS	An attack that floods a system to make it unavailable to users.	47,513	11,878
Mirai	Malware that infects IoT devices, creating botnets for launching large-scale DDoS attacks.	28,302	7075
MITM ARP Spoofing	Intercepting communication by tricking devices into sending data through the attacker's system.	332,247	83,062
Normal	Normal traffic.	32,058	8015
Scan	Probing a network or system to identify open ports, services, or vulnerabilities.	60,212	15,053

3.1.5. UAV IDS

The UAV IDS dataset is a specialized intrusion detection dataset designed to address security threats targeting unmanned aerial vehicles (UAVs). It focuses on two primary types of attacks: GPS spoofing and GPS jamming, both of which represent critical threats to UAV navigation and operational integrity. This focus on UAV-specific vulnerabilities makes the dataset particularly suitable for developing intrusion detection systems (IDS) aimed at protecting UAVs from cyber and electronic interference [40]. By including both attack data and benign flight data, the dataset offers a comprehensive foundation for training models to distinguish between normal and malicious UAV behavior highlighting the need to AI-based detection systems [41].

It contains three CSV files of GPS jamming, GPS spoofing, and benign flight with a total of 10,067 records. As no predefined split exists for training and testing, the total data are combined and divided using stratified train/test split to ensure class distribution in evaluating both types of attacks. The targeted nature and real-world relevance of the UAV IDS dataset make it a valuable contribution to researchers and developers working on UAV security and resilience against GPS-based attacks [42]. The attack categories presented in this dataset are shown in Table 6.

Table 6. UAV IDS [40] class descriptions and data distribution.

Class	Description	No. of Training	No. of Testing
Benign	Normal traffic.	4459	3650
Jamming	Disrupting communication signals by overwhelming them with interference or noise.	803	657
Spoofing	Falsifying data or signals to impersonate a legitimate source and deceive a target.	274	224

3.2. Data Preprocessing

We utilized MinMaxScaler to normalize numerical data and a one-hot encoder to process categorical features for consistency across all datasets. MinMaxScaler adjusts numerical values to a uniform range, between 0 and 1, ensuring that all features have comparable scales. This prevents features with larger magnitudes from dominating the learning process and preserves their relative differences. The one-hot encoder is used to convert categorical variables into a binary format where each category is represented by a distinct column. This encoding avoids implying any ordinal relationship between categories, ensuring that the model treats each category as equally distinct.

By applying MinMaxScaler and one-hot encoding, we ensured a standardized and consistent preprocessing pipeline. It also allows the model to interpret and learn from the data effectively. For MinMaxScaler and one-hot encoding in our preprocessing pipeline, it is common in network intrusion detection research to one-hot encode categorical fields and normalize numerical features (via min–max scaling) as part of basic data preparation. Several studies treat these steps as standard preprocessing, e.g., matrix formation in [43] and data preprocessing in [44].

We chose five datasets to demonstrate that our DNN architecture with vanilla KD framework works consistently well across a wide range of IDS scenarios in three industry-standard network benchmarks (NSL-KDD, UNSW-NB15, CIC-IDS2017), an IoT-focused dataset (IoTID20), and a UAV-specific GPS spoofing and jamming dataset. By using the same architecture and KD settings on each, we show that significant parameter reduction and faster inference speed-up are not artifacts of any single dataset, but it reflects a truly cross-domain capability that practitioners can rely on for both general-purpose and specialized, resource-constrained deployments.

3.3. Algorithms

3.3.1. Deep Neural Networks (DNN)

A deep neural network (DNN) is a type of artificial neural network (ANN) forming multiple hidden layers between the input and output layers. Modeled after the human brain, DNNs are designed to learn complex patterns from large datasets to make them effective for applications that require the identification of non-linear relationships. This capability is particularly valuable for anomaly-based intrusion detection systems (IDS), where distinguishing between normal and malicious behavior often involves recognizing subtle deviations in network traffic [45].

A DNN consists of the following components:

1. Input layer—Accepts input features
$$\mathbf{x} = [x_1, x_2, \ldots, x_n], \tag{1}$$
where n is the number of features.
2. Hidden layer—erforms transformations through neurons. Each neuron applies a linear transformation followed by a non-linear activation function (e.g., ReLU, sigmoid).
3. Output layer—Produces predictions $\hat{y}$.

For each layer l, the output $\mathbf{h}^{(l)}$ is computed as

$$\mathbf{h}^{(l)} = f\big(\mathbf{W}^{(l)}\,\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\big), \tag{2}$$

where:

- $\mathbf{W}^{(l)}$ is the weight matrix of layer l,
- $\mathbf{b}^{(l)}$ is the bias vector,
- $f(\cdot)$ is a non-linear activation function (ReLU).

The final output $\hat{y}$ is often passed through a softmax function for classification:

$$\hat{y} = \text{softmax}\big(\mathbf{W}^{(l+1)}\,\mathbf{h}^{(l+1)} + \mathbf{b}^{(l+1)}\big), \tag{3}$$

where L is the number of layers in the network.

In anomaly detection, the model learns to minimize a loss function $\mathcal{L}(\theta)$, such as the cross-entropy loss:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{m} \Big[y_i \log(\hat{y}_i) + (1 - y_i) \log\big(1 - \hat{y}_i\big) \Big], \tag{4}$$

where

- m is the number of samples,
- y_i is the true label (0 for normal, 1 for anomaly),
- $\hat{y}_i$ is the predicted probability of an anomaly.

The optimization of $\theta = \{\mathbf{W}, \mathbf{b}\}$ is performed using Adam.

DNNs are especially promising for anomaly-based IDS due to their ability to handle high-dimensional and complex network traffic data through their hierarchical structure, DNNs learn meaningful features automatically from the training data. This eliminates the need for manual feature engineering [46]. Furthermore, DNNs employ non-linear activation functions to model irregular patterns in network traffic, enabling detection of deviations from normal behavior. As a result, the use of DNNs can effectively detect unseen attack types and novel threats. These attributes make DNNs a robust choice for anomaly-based IDS, offering adaptive and scalable solutions to modern cyber threats [46,47]. We use the ADAM optimizer for training both teacher and student models due to its adaptive learning rate and efficient convergence properties [48], which are especially beneficial in optimizing deep neural networks on large and imbalanced intrusion detection datasets.

3.3.2. Knowledge Distillation (KD)

KD is one technique of model compression which knowledge from an underlying complex large model referred to as the teacher model, can be transferred to a much smaller model, called the student model [14]. The student model potentially achieves equivalent performance to the teacher model and therefore is suited for anomaly-based IDS in resource-constrained environments. The process relies on soft labels, the probability distributions generated by the teacher model, to guide student models at training. Such soft labels contain subtle information about class relationships, even when the predictions are wrong, helping the student model capture a better generalization. This approach allows the student model to learn both from the ground truth labels and the richer, softer probabilities of the teacher model [11].

The student model learns from the teacher by minimizing a combined loss function that includes the standard cross-entropy loss on the true labels and a distillation loss based on Kullback–Leibler divergence between the teacher's and student's softmax outputs, scaled by a temperature **T**. The soft outputs from the teacher provide richer class relationship information, which helps the student generalize better. This process enables the student to mimic the teacher's behavior as well as maintains a lightweight architecture.

We define:

- T: the *temperature parameter* that controls the softness of the probability distribution.
- p_i: the soft probability (output by the **teacher** model) for class i.
- q_i: the soft probability (output by the **student** model) for class i.

The soft probabilities are computed using the softmax function with temperature T:

$$q_i = \frac{\exp\big(z_i/T\big)}{\sum_j \exp\big(z_j/T\big)}. \tag{5}$$

where $\exp(z_i/T)$ represents the scaled exponent of the logit for class i (pre-softmax output), the denominator $\sum_j \exp(z_j/T)$ sums over the exponentials of all class logits to normalize the outputs into a probability distribution, and T is the temperature parameter controlling the softness of the output distribution.

The distillation loss $\mathcal{L}_{\text{distill}}$ measures the difference between the soft probabilities of the teacher and student models using Kullback–Leibler (KL) divergence:

$$\mathcal{L}_{\text{distill}} = T^2 \sum_i p_i \log\left(\frac{q_i}{p_i}\right). \tag{6}$$

The overall objective for training the student model combines the distillation loss and the standard cross-entropy loss L_{CE} (computed using the true class labels $\mathbf{y}$):

$$L_{\text{total}} = \alpha\, L_{\text{distill}} + (1 - \alpha)\, L_{\text{CE}}, \tag{7}$$

where:

- α is a hyperparameter balancing the two losses. It is a weighting factor that balances the distillation and cross-entropy losses.
- L_{CE} is defined as follows:
$$L_{\text{CE}} = -\sum_i y_i \log(p_i). \tag{8}$$

KD has great potential for anomaly-based IDSs due to the following reasons: first, knowledge distillation can enable lightweight model deployment that becomes crucial during operation on resource-constrained devices, including IoT nodes or UAVs. Second, the generalization achieved by the student model by learning subtle patterns from the teacher model improves significantly. Also, the improved generalization ability of the student model would help in detecting unseen attacks or novel attacks, which might not occur in only student model alone without KD.

In this study, the teacher–student model represents a knowledge distillation framework designed to balance accuracy and efficiency in deep learning tasks. The teacher model is a larger and more complex neural network composed of four dense layers with 256, 128, 64, and 32 units, respectively. Each layer uses the ReLU activation function to capture complex and hierarchical patterns in the data. The output layer corresponds to the number of classes in the target variable for the output as per multi-class classification. This architecture enables the teacher model to extract rich, detailed representations of the input data. The student model, in contrast, is a smaller and simplified version of the teacher. It consists of two dense layers with 32 and 16 units, respectively, also using ReLU activation. The student's output layer matches that of the teacher, ensuring it can perform the same classification task. The architecture used in this study is common in KD IDS. For example, Jeong et al. [49] show a two-layer student MLP matching a deeper teacher, Wu et al. [50] use a two-layer FC net (64→32) in an IDS. By distilling the knowledge from the teacher's output, which provides soft probabilities showing class relationships, the student model approximates the teacher's performance while significantly reducing computational and memory demands. Figure 1 describes the knowledge distillation structure we implemented in this study.

The knowledge distillation technique involves two key parameters: α (alpha) and T (temperature). For consistency across all experiments, we set $\alpha = 0.4$ and $T = 3$. For Repeatability, we fixed the random seed to 42 for NumPy, TensorFlow/Keras, and for every train_test_split in cross validation calls (random_state = 42). This guarantees identical data splits and weight initialization across different runs. Many recent IDS and related works adopt similar values for these parameters. For example, Ren et al. [20] distill a large LSTM model into a small 2-layer feedforward network and set the KD balance factor (alpha) to

0.45 for all their experiments. Likewise, Yang et al. [30] propose an intrusion detection model (LNet-SKD) and report that the best performance occurs with temperature, T = 3. In both cases, these values are chosen as fixed defaults, not per-dataset tuning.

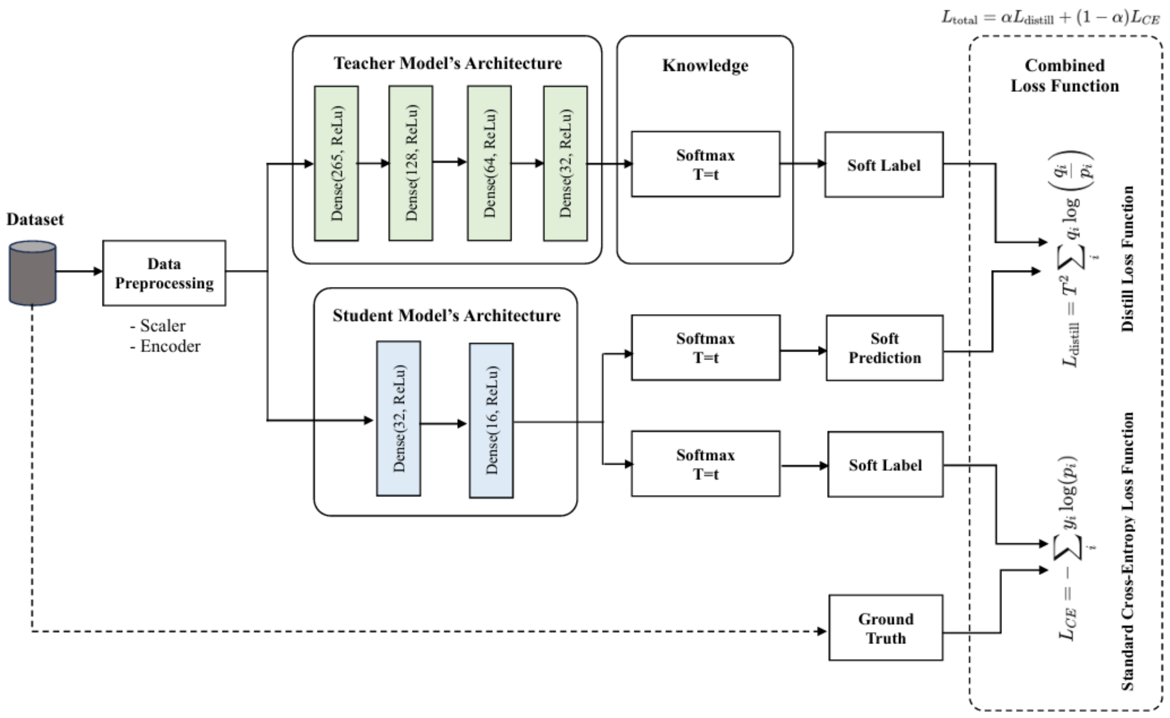

Figure 1. Knowledge distillation from DNN dense (256, 128, 64, 32) to (32, 16) structure.

4. Result

To evaluate the performance of models in a multi-class classification setting, we use seven metrics, accuracy, precision, recall, F1 score, ROC, AUC, and inference speed, as shown in Table 7. In the case of multi-class classification, these metrics are calculated for each class individually and then averaged across all classes to provide a comprehensive assessment.

Table 7. Evaluation Metrics used in this Study.

Metric	Formula	Description
Accuracy	$\frac{\text{Total Correct Predictions}}{\text{Total Predictions}}$	Measures how often the model predicts correctly for all classes.
Precision (weighted averaged)	$\frac{\sum \text{TP}}{\sum(\text{TP}+\text{FP})}$	Number of positive predictions that are actually correct.
Recall (weighted averaged)	$\frac{\sum \text{TP}}{\sum(\text{TP}+\text{FN})}$	Number of actual positives that the model truly identifies.
F1 Score (Weighted-Averaged)	$\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$	Balances precision and recall; prioritizes classes with more true instances.
True positive rate (TPR)	$\frac{\text{TP}}{\text{TP}+\text{FN}}$	Fraction of positives correctly identified.
False positive rate (FPR)	$\frac{\text{FP}}{\text{FP}+\text{TN}}$	Fraction of negatives incorrectly identified as positives.
AUC (Area Under Curve)	–	Measures the area under the ROC curve; higher AUC means better performance.
Inference Speed	$\frac{\text{Total Inference Time}}{\text{Number of Instances}}$	Measures how fast the model processes each instance.

Notation: TP: true positives, TN: true negatives, FP: false positives, FN: false negatives.

It is worth noting that the inference time (second) is calculated as the average of ten runs under the same environment and settings, measured per one data entry (row), and multiplied with (10^{-5}). Changes in the number of parameters and inference speed are compared against the teacher model, while metrics such as accuracy, precision, recall, and F1 score are compared against the student model without KD to assess improvements on the knowledge transfer.

Hardware/software: All experiments ran on a workstation (Windows 10, Intel Core i9 9900 3.10 GHz, 32 GB of RAM, and Nvidia RTX 2070 Super).

4.1. NSL-KDD

The results from the NSL-KDD dataset (Table 8) demonstrate the trade-off between model complexity and performance, as well as the effectiveness of KD in enhancing lightweight models. The teacher model, a deep neural network with 224,657 parameters, achieves the highest overall performance, including an accuracy of 77.22%, precision of 82.16%, and F1 score of 73.05%. However, its large size results in a slower inference speed of 2.935×10^{-5} s per instance, which limits its practicality in real-time applications. The student model without KD, significantly smaller with only 13,649 parameters (a 93.93% reduction), achieves a slightly lower accuracy of 75.96% and a modest drop in F1 score to 70.97%, despite maintaining a comparable precision of 82.06%. Its inference speed improves to 2.630×10^{-5} s, making it better suited for deployment, but the reduced F1 score indicates diminished recall, especially for minority classes.

After applying KD, the student model shows consistent improvements across all metrics while maintaining a compact architecture and fast inference. Accuracy increases to 76.73% and F1 score to 71.83%. Inference speed further improves to 2.599×10^{-5} s, a reduction of 11.45% compared to the teacher. Analysis of the confusion matrices and ROC curves (Figures 2 and 3) provides additional insight. The teacher model achieves strong class separation, with AUC values of 0.96 for DoS and U2R, 0.93 for probe, and 0.91 for normal. The KD-enhanced student closely follows, reaching AUC values of 0.98 for U2R and 0.91 for DoS, while reducing misclassifications such as Probe incorrectly labeled as DoS (185 vs. 253 without KD). In contrast, the non-distilled student performs worse on rare classes like R2L and Probe, where AUC drops to 0.70 and 0.89, respectively. These findings highlight KD's role in transferring generalization capability from the teacher, improving the student's performance even under class imbalance. However, this dataset suffers from severe class imbalance in training; as such, the models rarely predict R2L and U2R correctly without leveraging other techniques.

Table 8. Comparison of teacher and student models on the NSL-KDD dataset.

Model's Architecture	**No. of Parameters**	**Accuracy (%)**	**Precision (%)**	**F1 Score (%)**	**Inference Time**
Deep Neural Network (Teacher)	224,657	77.22	82.16	73.05	2.935
Shallow Neural Network (Student without KD)	13,649	75.96	82.06	70.97	2.630
Shallow Neural Network (Student with KD)	13,649 (93.93%)	76.73 (+0.77%)	82.62 (+0.56%)	71.83 (+0.86%)	2.599 (−11.45%)

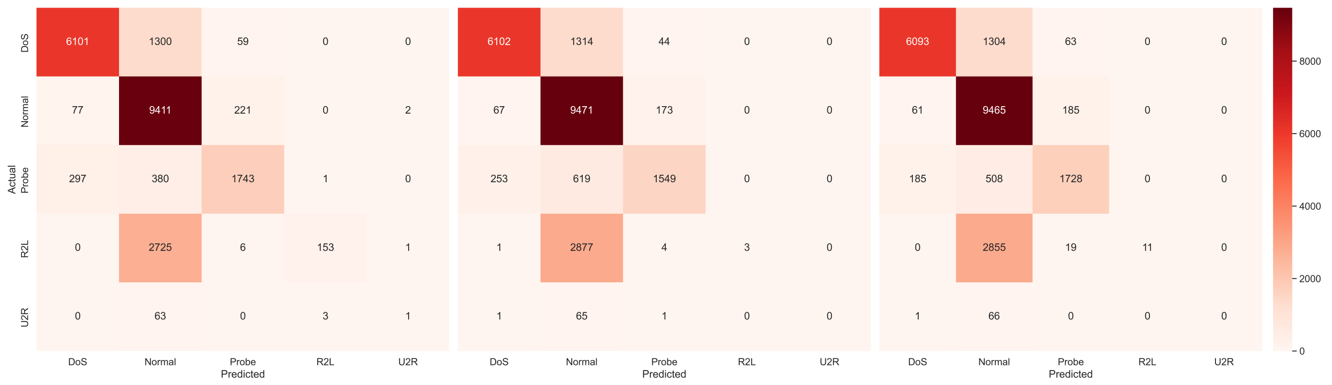

Figure 2. Confusion matrix of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on NSL-KDD.

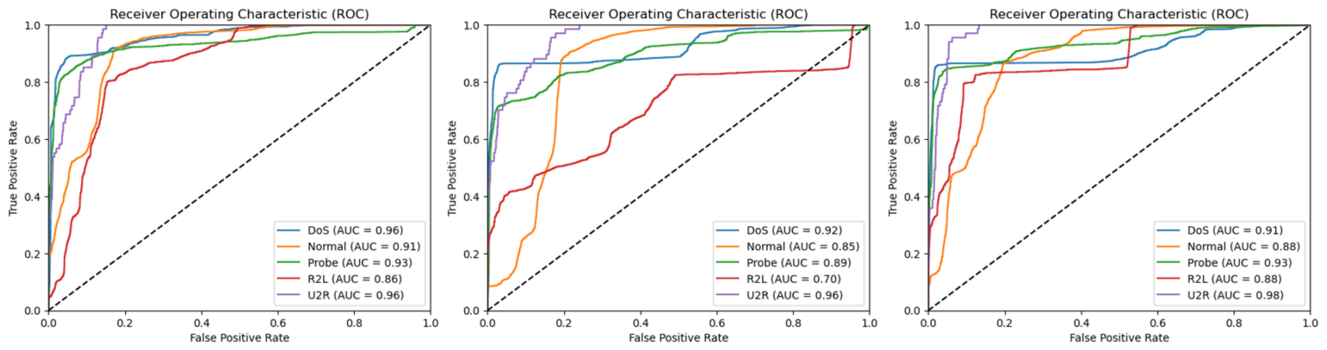

Figure 3. ROC curve of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on NSL-KDD.

4.2. UNSW-NB15

The UNSW-NB15 results (Table 9) show a similar pattern to NSL-KDD, where the teacher model delivers the best overall performance at the cost of computational efficiency. With 280,448 parameters, the teacher model achieves 75.45% accuracy, 81.12% precision, and an F1 score of 76.40%, reflecting its capacity to capture complex patterns in the data. However, this comes with a slower inference speed of 2.722×10^{-5} s. The student model without KD, while significantly more efficient with only 20,816 parameters, sees a drop in performance—achieving 69.66% accuracy and a lower F1 score of 71.10%. Despite this, it runs faster (2.502×10^{-5} s), suggesting that the smaller model is able to maintain confident predictions, though it may miss more true positives due to reduced capacity.

Introducing KD narrows this performance gap. The distilled student model reaches 75.78% accuracy and 76.10% F1 score—improvements of +6.12% and +5.00% percentage points, respectively, over the non-distilled version, while retaining a fast inference speed of 2.451×10^{-5} s. The confusion matrices and ROC curves (Figures 4 and 5) further confirm this improvement: the KD-enhanced student exhibits fewer misclassifications across multiple classes and achieves AUC values close to those of the teacher model. The non-distilled student, on the other hand, shows a noticeable drop in AUC, especially for rare attack types like worms and Shellcode. These results suggest that KD successfully transfers the teacher's ability to generalize across both majority and minority classes. However, performance on underrepresented classes remains an area where further improvements are needed.

Table 9. Comparison of teacher and student models on the UNSW-NB15 dataset.

Model's Architecture	No. of Parameters	Accuracy (%)	Precision (%)	F1 Score (%)	Inference Time
Deep Neural Network (Teacher)	280,448	75.45	81.12	76.40	2.722
Shallow Neural Network (Student without KD)	20,816	69.66	78.12	71.10	2.502
Shallow Neural Network (Student with KD)	20,816 (−92.57%)	75.78 (+6.12%)	80.98 (+2.86%)	76.10 (+5.00%)	2.451 (−9.96%)

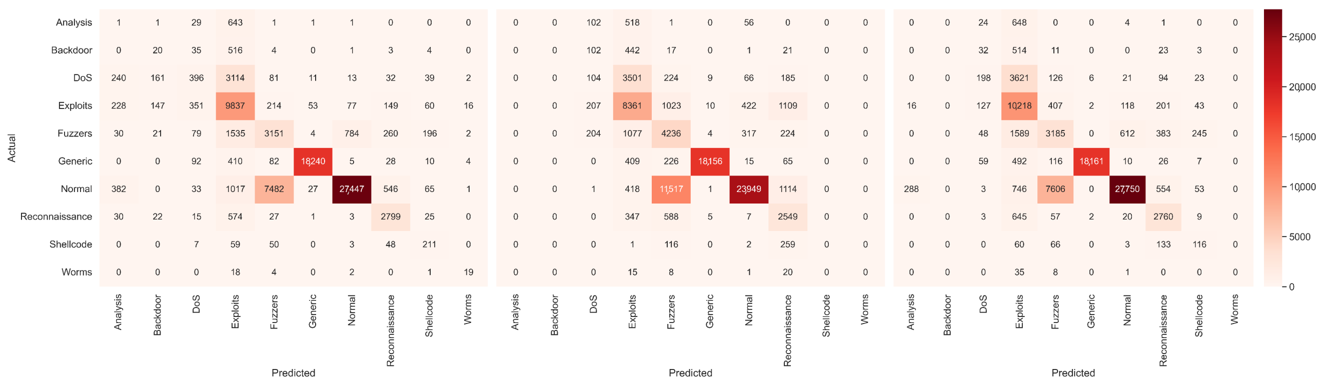

Figure 4. Confusion matrix of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on UNSW-NB15.

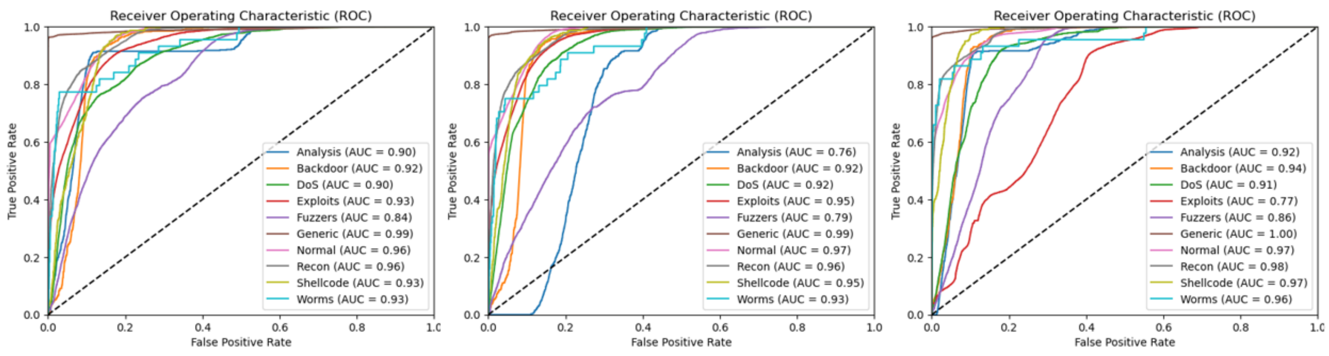

Figure 5. ROC curve of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on UNSW-NB15.

4.3. CIC-IDS2017

The CIC-IDS2017 results (Table 10) further confirm the pattern observed in the previous datasets that the teacher model achieves the highest accuracy (98.22%) and F1 score (98.20%) due to its large capacity and deeper architecture. However, this comes at the cost of computational efficiency, with a slower inference speed of 2.602×10^{-5} s. The student model without KD, while significantly smaller with 9,527 parameters, delivers slightly lower performance at 97.88% accuracy and 97.85% F1 score—but improves inference speed to 2.398×10^{-5} s. The minimal drop in performance alongside faster execution highlights the efficiency of the smaller model in handling large data.

After applying KD, the student model shows a marginal yet consistent improvement in all key metrics: accuracy increases to 97.88% and F1 score to 97.85%. Inference speed improves slightly to 2.398×10^{-5} s. These results suggest that KD still provides a benefit, even when the baseline student model already performs strongly. The confusion matrices (Figure 6) indicate that KD helps further reduce misclassifications in certain classes, though the differences are less pronounced than in the other datasets. The ROC curves (Figure 7) show high AUC values across all classes for both the teacher and KD-enhanced student, confirming that the models are effective at distinguishing between attack types and normal traffic. Overall, these findings show that while the impact of KD is subtler in highly separable data, it remains useful for fine-tuning lightweight models and preserving performance under resource constraints.

Table 10. Comparison of teacher and student models on the CIC-IDS2017 dataset.

Model's Architecture	No. of Parameters	Accuracy (%)	Precision (%)	F1 Score (%)	Inference Time
Deep Neural Network (Teacher)	191,063	98.22	98.30	98.20	2.602
Shallow Neural Network (Student without KD)	9527	97.81	97.97	97.79	2.382
Shallow Neural Network (Student with KD)	9527 (−95.01%)	97.88 (+0.07%)	97.95 (+0.02%)	97.85 (+0.06%)	2.398 (−7.84%)

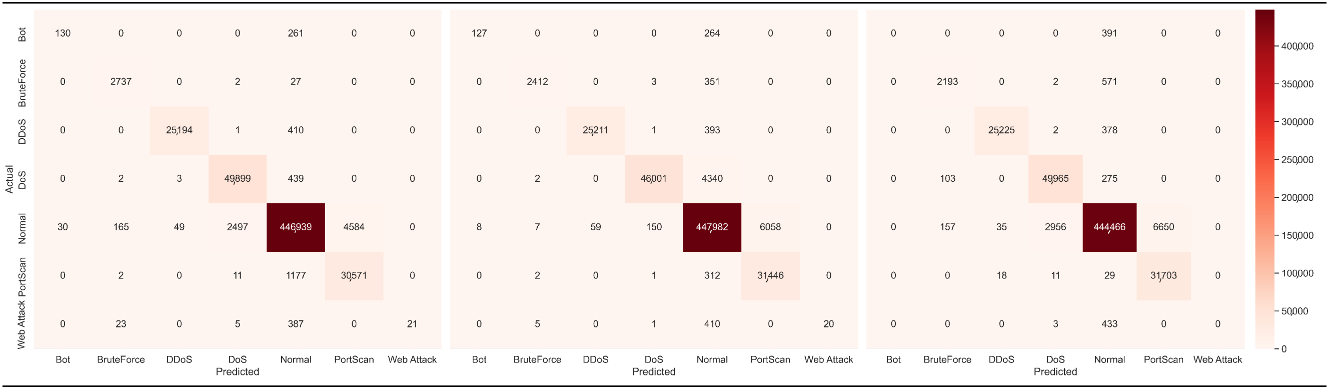

Figure 6. Confusion matrix of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on CICIDS2017.

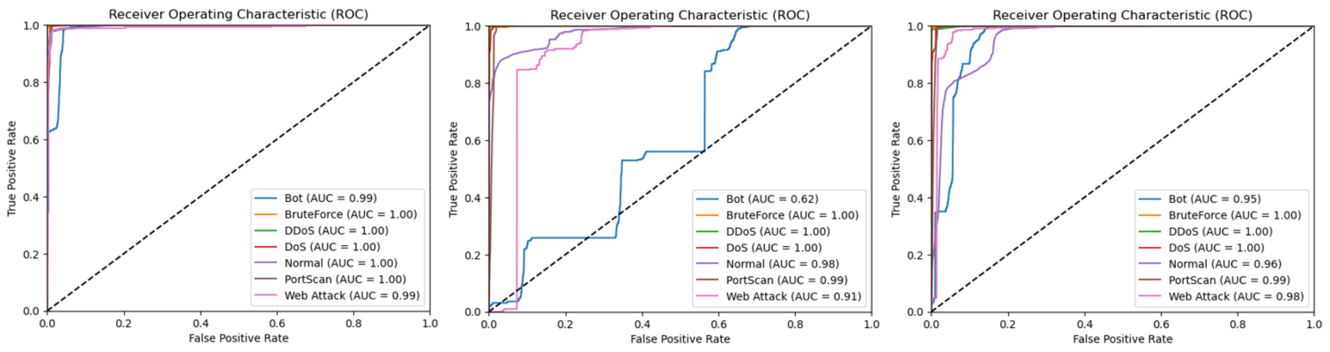

Figure 7. ROC curve of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on CICIDS2017.

4.4. IoTID20

The IoTID20 dataset results (Table 11) reveal the same performance–efficiency trade-off seen in previous experiments but under more challenging data conditions. The teacher model with its complexity, achieves relatively lower accuracy (92.95%) and F1 score (93.13%) compared to results on other datasets. This suggests that the IoTID20 dataset, with its device-specific patterns and potential noise or imbalance, poses greater difficulty for even large models. The student model without KD, though more efficient with just 9521 parameters, experiences a further drop in F1 score to 86.20% and accuracy to 87.15%, reflecting the limits of smaller architectures when learning from limited or less generalizable data. Inference speed improves to 2.386×10^{-5} s compared to the teacher's 2.651×10^{-5}, demonstrating the student model's computational advantage.

When KD is applied, the student model shows a clear performance gain, increasing its accuracy to 93.24% and F1 score to 93.13%, while slightly improving inference speed to 2.462×10^{-5} s. Although these improvements are modest, they demonstrate that KD helps the student model better approximate the teacher's generalization ability, even under the more complex and potentially noisier conditions of IoT data. The confusion matrices (Figure 8) and ROC curves (Figure 9) show that the KD-enhanced student achieves better separation across most classes compared to the non-distilled variant, particularly for attack categories with fewer samples such as Mirai and MITM ARP Spoofing. These findings support KD as a practical technique for improving lightweight model performance in real-world, resource-constrained IoT applications.

Table 11. Comparison of teacher and student models on the IoTID20 dataset.

Model's Architecture	No. of Parameters	Accuracy (%)	Precision (%)	F1 Score (%)	Inference Time
Deep Neural Network (Teacher)	191,633	92.95	94.04	93.13	2.651
Shallow Neural Network (Student without KD)	9521	87.15	88.51	86.20	2.386
Shallow Neural Network (Student with KD)	9521 (−95.03%)	93.24 (+6.09%)	93.42 (+4.91%)	93.13 (+6.93%)	2.462 (−7.13%)

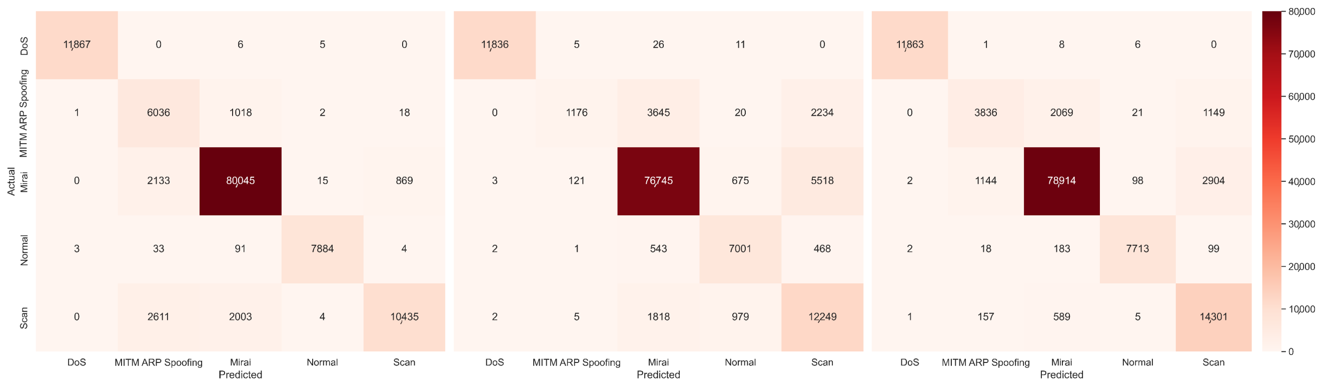

Figure 8. Confusion matrix of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on IoTID20.

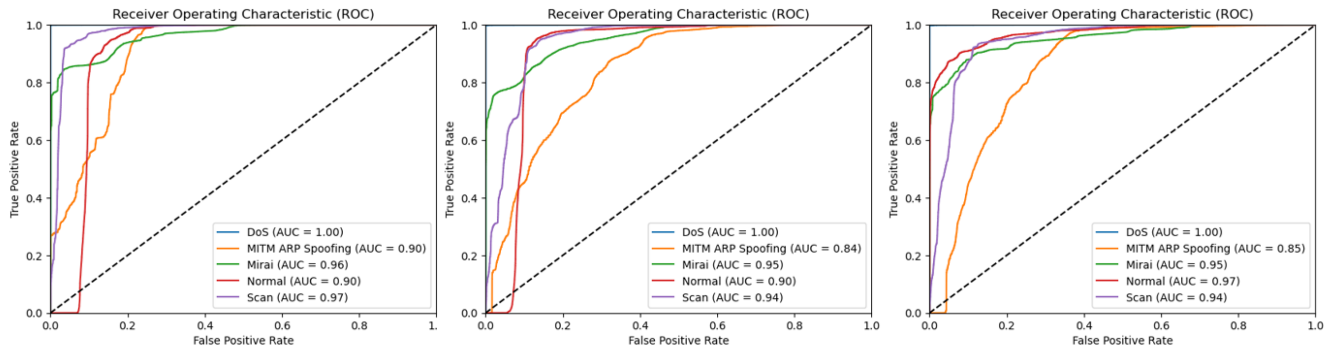

Figure 9. ROC curve of teacher (**left**), student without KD (**middle**), student with KD (**right**) models on IoTID20.

4.5. UAV IDS

The UAV IDS results (Table 12) reflect a similar pattern to other datasets, with the teacher model achieving the strongest performance overall but at a higher computational cost. The teacher model reaches an accuracy and F1 score of 99.93%, reflecting its ability to capture complex patterns in UAV network traffic. However, its inference speed is relatively slow at 2.650×10^{-5} s, due to the model's size (194,507 parameters). The student model without KD, despite having only 9803 parameters, performs well with an accuracy of 97.97% and F1 score of 97.86%, and a faster inference speed of 2.386×10^{-5} s. This suggests that even without KD, the student model is capable of learning dominant patterns in the UAV IDS dataset, although it sacrifices a small degree of generalization and recall.

After applying KD, the student model sees further gains in performance, achieving 98.63% accuracy and an F1 score of 98.59%, while also improving inference speed slightly to 2.462×10^{-5} s. These results indicate that KD effectively bridges the performance gap, enabling the student model to approximate the teacher's performance more closely without obviously increasing its computational footprint. The confusion matrices and ROC curves (Figures 10 and 11) show that the KD-enhanced student achieves improved class separability and reduced misclassification rates, particularly in detecting UAV-specific attacks such as GPS jamming and spoofing. Although the dataset is relatively small, the improvements through KD emphasize its value even in less noisy or skewed scenarios. These findings support the feasibility of using KD to deploy accurate, real-time UAV intrusion detection systems on resource-constrained platforms.

Table 12. Comparison of teacher and student models on the UAV IDS dataset.

Model's Architecture	No. of Parameters	Accuracy (%)	Precision (%)	F1 Score (%)	Inference Time
Deep Neural Network (Teacher)	194,507	99.93	99.93	99.93	2.650
Shallow Neural Network (Student without KD)	9803	97.97	98.02	97.86	2.386
Shallow Neural Network (Student with KD)	9803 (−94.96%)	98.63 (+0.66%)	98.65 (+0.63%)	98.59 (+0.73%)	2.462 (−7.09%)

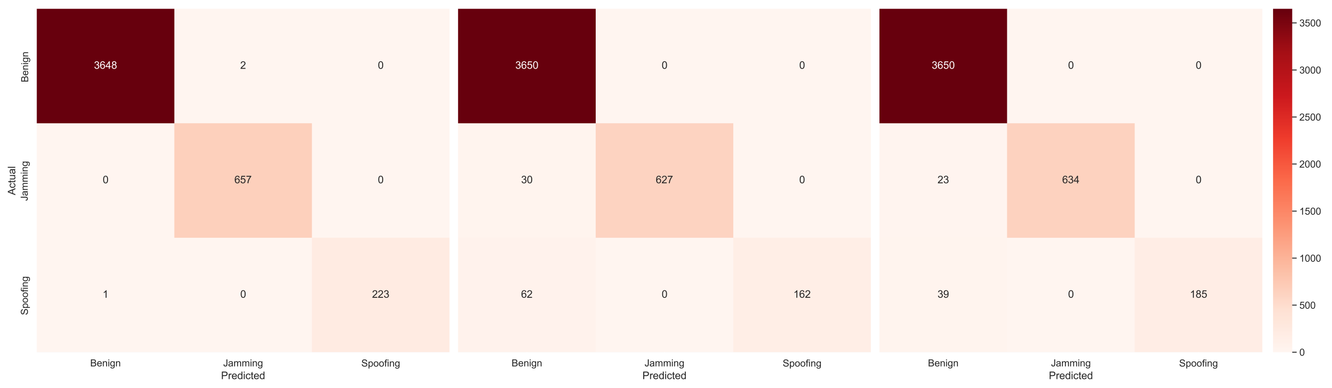

Figure 10. Confusion matrix of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on UAV IDS.

Additionally, we provided a table for statistical analysis (confidence intervals) as shown in Table 13. Across the five datasets, we observe that for NSL-KDD, UNSW-NB15, and CIC-IDS2017, the 95% confidence intervals for the teacher and distilled student overlap almost completely, indicating that the tiny accuracy differences (less than 0.1%) lie within random variation and are not statistically significant. In contrast, on IoTID20 the distilled student's interval (93.10–93.38%) sits entirely above the undistilled student's (86.96–87.34%), showing a significant gain from KD. For the smaller dataset such as UAV IDS, both student models remain significantly below the teacher despite KD recovering some performance. Thus, only on IoTID20 with KD yields a statistically significant accuracy boost across all datasets. On the other hand, KD consistently delivers 92–95% parameter reduction and up to 11% faster inference. This measure makes it valuable beyond accuracy gain alone.

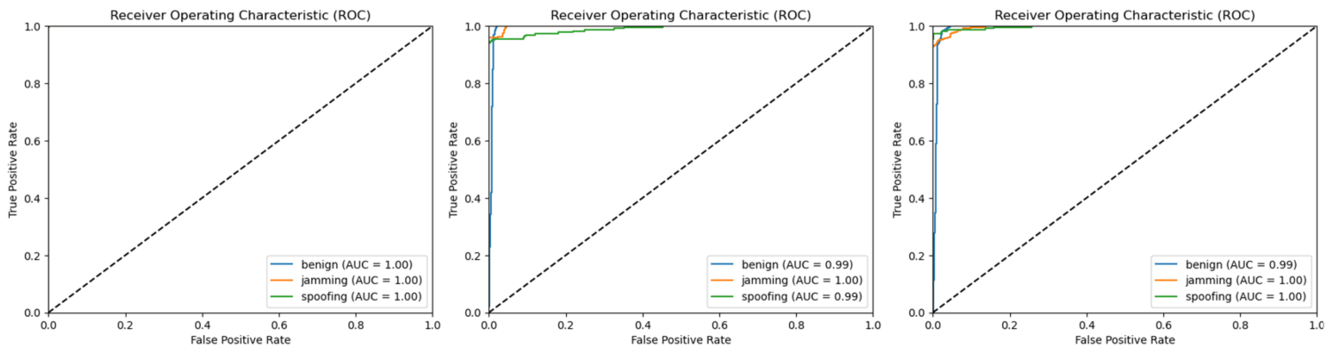

Figure 11. ROC curve of teacher (**left**), student without KD (**middle**), and student with KD (**right**) models on UAV IDS.

Table 13. Statistical summary of model results (95% confidence interval).

Dataset	# Test Samples	Model	Accuracy (%)	95% CI Lower (%)	95% CI Upper (%)
NSL-KDD	22,544	Teacher	77.22	76.67	77.77
		Student without KD	75.96	75.40	76.52
		Student with KD	76.73	76.18	77.28
UNSW-NB15	82,332	Teacher	75.45	75.16	75.74
		Student without KD	69.66	69.35	69.97
		Student with KD	75.78	75.49	76.07
CIC-IDS2017	565,566	Teacher	98.22	98.19	98.25
		Student without KD	97.81	97.77	97.85
		Student with KD	97.88	97.84	97.92
IoTID20	125,083	Teacher	92.95	92.81	93.09
		Student without KD	87.15	86.96	87.34
		Student with KD	93.24	93.10	93.38
UAV IDS	4531	Teacher	99.93	99.86	100.00
		Student without KD	97.97	97.56	98.38
		Student with KD	98.63	98.29	98.97

4.6. Discussion

The IoTID20 results show that the distilled student model delivers exceptional detection for the most prevalent attack types, e.g., DoS, Mirai, normal traffic, and scan, all with recalls above 96% and similarly high precision, driving the overall misclassification rate below 1%. False positives on benign traffic remain under 0.05%, so operators will not be overwhelmed by false alerts. While the rarer ARP-spoofing class shows somewhat lower recall, it represents under 5% of total flows, so its impact on overall IDS performance is relatively minimal.

The UAV IDS results demonstrate that the model produces zero false positives on benign flights which is vital to avoid triggering unnecessary safe-landing procedures. But it misses about 3.5% of jamming and 17% of spoofing events. This trade-off (almost no false alarms versus a modest miss rate) reflects the operational reality of small drones that it is generally more critical to avoid mission-halting false alerts, but reducing undetected attacks remains an important goal for future work.

Inference latency is a critical bottleneck on resource-limited IoT and UAV controllers, and our KD framework reduces per-instance inference time by 7–11%, an improvement that can meaningfully shorten detection delays and enable faster countermeasures in flight. Likewise, reducing model parameters by over 90% suggests substantial energy savings potentially extending UAV mission time or allowing longer continuous IDS operation. These gains also support quicker system recovery or safe-landing procedures by narrowing the window during which malicious inputs can mislead the vehicle. Although KD may introduce its own security considerations (such as inherited adversarial weaknesses), a detailed assessment of these risks is beyond the scope of this study. It is worth noting that recent work shows that adversarial vulnerabilities can persist through knowledge distillation and across different application domains. Goldblum et al. [51] found that a student model inherits both its teacher's robustness and its susceptibility to adversarial examples, meaning that any blind spots in the teacher can be transferred to the student. Sheatsley et al. [52] further demonstrated that carefully crafted adversarial network traffic can evade IDS models with success rates exceeding 95%, even when only permissible features are modified under realistic constraints. Therefore, deploying KD-based IDS should include strategies to investigate and mitigate potential adversarial vulnerabilities.

In IoT deployments, the lower recall on rare ARP-spoofing attacks means that a small number of man-in-the-middle intrusions could be unnoticed, but since these flows account for under 5% of traffic and downstream monitoring tools can catch anomalous behavior, the overall network risk remains manageable. In UAV operations, missing 3% of spoofing and 17% jamming events could allow brief navigation errors, yet maintaining zero false positives ensures that emergency landings are not triggered unnecessarily—preserving mission continuity. Balancing these trade-offs in practice means implementing the KD-based IDS can help detect rare attacks without sacrificing significant operational capabilities.

5. Conclusions

This paper implements a lightweight intrusion detection framework based on knowledge distillation (KD) to enhance the performance of compact neural networks in resource-constrained environments, such as IoT and UAV systems. The primary contribution is that we systematically evaluated a KD-based approach that transfers knowledge from a large, high-performing teacher model to a smaller student model. This enables improved detection capabilities while maintaining computational efficiency. Experimental results across five benchmark datasets (NSL-KDD, UNSW-NB15, CIC-IDS2017, IoTID20, and UAV IDS) show that KD consistently improves the student model's performance. For example, in the IoTID20 dataset, the distilled student improved the F1 score from 86.20% to 93.13%, while reducing inference time by 7.13% compared to the teacher. Similarly, in the UAV IDS dataset, the KD-enhanced student reached 98.59% F1 score, just 1.34 percentage points below the teacher while increasing 7.09% faster inference. Across all datasets, the student model with KD achieved F1 score improvements up to 6.93%, and AUC values that closely matched those of the teacher model.

These results confirm that knowledge distillation offers an effective trade-off between accuracy and efficiency, making it suitable for real-time intrusion detection especially in IoT and UAV security. The framework not only reduces model complexity by over 93% in parameter count but also delivers competitive detection performance. As IoT and UAV networks expand, this study demonstrates KD's potential to serve as a foundational technology for deploying IDS in next-generation edge computing strategies, ensuring secure, efficient, and scalable operations.

Future work can include the following directions (1) Investigating additional strategies, such as handling class imbalance, to further improve performance on rare attack types and evolving threats. (2) Incorporating adversarial robustness evaluation into the framework, particularly given the security-critical nature of domains like UAVs, where adversarial attacks could pose serious risks to operational safety and system reliability. (3) Comparing knowledge distillation with other popular model compression techniques such as pruning or quantization. This will provide a more comprehensive understanding of trade-offs in accuracy and deployment feasibility in constrained environments. Such a holistic benchmark will help guide the selection of optimal compression strategies for specific real-world applications like IoT and UAV security.

Author Contributions: Conceptualization, M.T.; methodology, T.W.; software, T.W.; validation, T.W.; formal analysis, T.W.; data curation, T.W.; writing—original draft, T.W.; writing—review and editing, T.W.; visualization, T.W.; supervision, M.T.; project administration, M.T.; funding acquisition, M.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Royal Thai Air Force Cyber Security Research Program.

Data Availability Statement: The dataset used in this research is publicly available.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Abbreviation	Definition
AB-TRAP	Attack Bonafide Train RealizAtion and Performance
AUC	Area Under the Curve
CNN	Convolutional Neural Network
CPS	Cyber Physical System
DNN	Deep Neural Network
FCN	Fully-Connected Network
FFCNN	Feedforward Convolutional Neural Network
GAN	Generative Adversarial Network
GPS	Global Positioning System
IDS	Intrusion Detection System
IMU	Inertial Measurement Unit
IoT	Internet of Things
KD	Knowledge Distillation
KDDT	Knowledge Distillation-Empowered Digital Twin for Anomaly Detection
LNet-SKD	Lightweight Intrusion Detection Approach based on Self-Knowledge Distillation
NFQUEUE	Netfilter QUEUE
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RSSI	Received Signal Strength Indicator
SSFL	Semi-Supervised Federated Learning
TBCLNN	Tied Block Convolution Lightweight Deep Neural Network
UAV	Unmanned Aerial Vehicle

References

1. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 1–22. [CrossRef]
2. Scarfone, K.; Mell, P. Guide to intrusion detection and prevention systems (idps). *NIST Spec. Publ.* **2007**, *800*, 94.

3. Hussain, A.; Sharif, H.; Rehman, F.; Kirn, H.; Sadiq, A.; Khan, M.S.; Riaz, A.; Ali, C.N.; Chandio, A.H. A systematic review of intrusion detection systems in internet of things using ML and DL. In Proceedings of the 2023 4th International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 17–18 March 2023; pp. 1–5.
4. Sommer, R.; Paxson, V. Outside the closed world: On using machine learning for network intrusion detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 16–19 May 2010; pp. 305–316.
5. Fernandes, G.; Rodrigues, J.J.; Carvalho, L.F.; Al-Muhtadi, J.F.; Proença, M.L. A comprehensive survey on network anomaly detection. *Telecommun. Syst.* **2019**, *70*, 447–489. [CrossRef]
6. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
7. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
8. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A deep learning approach to network intrusion detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
9. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* **2018**, *1*, 108–116.
10. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531. [CrossRef]
11. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]
12. Cheng, Y.; Wang, D.; Zhou, P.; Zhang, T. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Process. Mag.* **2018**, *35*, 126–136. [CrossRef]
13. Teerapittayanon, S.; McDanel, B.; Kung, H.T. Distributed deep neural networks over the cloud, the edge and end devices. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 328–339.
14. Wang, Z.; Li, Z.; He, D.; Chan, S. A lightweight approach for network intrusion detection in industrial cyber-physical systems based on knowledge distillation and deep metric learning. *Expert Syst. Appl.* **2022**, *206*, 117671. [CrossRef]
15. Li, Z.; Yao, W. A two stage lightweight approach for intrusion detection in Internet of Things. *Expert Syst. Appl.* **2024**, *257*, 124965. [CrossRef]
16. Wang, L.H.; Dai, Q.; Du, T.; Chen, L.f. Lightweight intrusion detection model based on CNN and knowledge distillation. *Appl. Soft Comput.* **2024**, *165*, 112118. [CrossRef]
17. Zhao, R.; Chen, Y.; Wang, Y.; Shi, Y.; Xue, Z. An efficient and lightweight approach for intrusion detection based on knowledge distillation. In Proceedings of the ICC 2021—IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6.
18. Chen, J.; Guo, Q.; Fu, Z.; Shang, Q.; Ma, H.; Wang, N. Semi-supervised Campus Network Intrusion Detection Based on Knowledge Distillation. In Proceedings of the 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 18–23 June 2023; pp. 1–7.
19. Zhu, S.; Xu, X.; Zhao, J.; Xiao, F. Lkd-stnn: A lightweight malicious traffic detection method for internet of things based on knowledge distillation. *IEEE Internet Things J.* **2023**, *11*, 6438–6453. [CrossRef]
20. Ren, Y.; Restivo, R.D.; Tan, W.; Wang, J.; Liu, Y.; Jiang, B.; Wang, H.; Song, H. Knowledge distillation-based GPS spoofing detection for small UAV. *Future Internet* **2023**, *15*, 389. [CrossRef]
21. Abbasi, M.; Shahraki, A.; Prieto, J.; Arrieta, A.G.; Corchado, J.M. Unleashing the potential of knowledge distillation for IoT traffic classification. *IEEE Trans. Mach. Learn. Commun. Netw.* **2024**, *2*, 221–239. [CrossRef]
22. Zhao, R.; Yang, L.; Wang, Y.; Xue, Z.; Gui, G.; Ohtsuki, T. A semi-supervised federated learning scheme via knowledge distillation for intrusion detection. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 2688–2693.
23. Shen, J.; Yang, W.; Chu, Z.; Fan, J.; Niyato, D.; Lam, K.Y. Effective intrusion detection in heterogeneous Internet-of-Things networks via ensemble knowledge distillation-based federated learning. In Proceedings of the ICC 2024—IEEE International Conference on Communications, Denver, CO, USA, 9–13 June 2024; pp. 2034–2039.
24. Quyen, N.H.; Duy, P.T.; Nguyen, N.T.; Khoa, N.H.; Pham, V.H. FedKD-IDS: A robust intrusion detection system using knowledge distillation-based semi-supervised federated learning and anti-poisoning attack mechanism. *Inf. Fusion* **2025**, *117*, 102807. [CrossRef]
25. Ravipati, R.D.; Abualkibash, M. Intrusion detection system classification using different machine learning algorithms on KDD-99 and NSL-KDD datasets-a review paper. *Int. J. Comput. Sci. Inf. Technol. (IJCSIT) Vol* **2019**, *11*, 65–80. [CrossRef]
26. Ali, T.; Eleyan, A.; Bejaoui, T.; Al-Khalidi, M. Lightweight Intrusion Detection System with GAN-Based Knowledge Distillation. In Proceedings of the 2024 International Conference on Smart Applications, Communications and Networking (SmartNets), Harrisonburg, VA, USA, 28–30 May 2024; pp. 1–7.
27. Bertoli, G.D.C.; Júnior, L.A.P.; Saotome, O.; Dos Santos, A.L.; Verri, F.A.N.; Marcondes, C.A.C.; Barbieri, S.; Rodrigues, M.S.; De Oliveira, J.M.P. An end-to-end framework for machine learning-based network intrusion detection system. *IEEE Access* **2021**, *9*, 106790–106805. [CrossRef]

28. Hassler, S.C.; Mughal, U.A.; Ismail, M. Cyber-physical intrusion detection system for unmanned aerial vehicles. *IEEE Trans. Intell. Transp. Syst.* **2023**, *25*, 6106–6117. [CrossRef]
29. Hadi, H.J.; Cao, Y.; Li, S.; Hu, Y.; Wang, J.; Wang, S. Real-time collaborative intrusion detection system in uav networks using deep learning. *IEEE Internet Things J.* **2024**, *11*, 33371–33391. [CrossRef]
30. Yang, S.; Zheng, X.; Xu, Z.; Wang, X. A lightweight approach for network intrusion detection based on self-knowledge distillation. In Proceedings of the ICC 2023—IEEE International Conference on Communications, Rome, Italy, 28 May–1 June 2023; pp. 3000–3005.
31. Wang, Z.; Zhou, R.; Yang, S.; He, D.; Chan, S. A Novel Lightweight IoT Intrusion Detection Model Based on Self-knowledge Distillation. *IEEE Internet Things J.* **2025**, *12*, 16912–16930. [CrossRef]
32. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
33. Wisanwanichthan, T.; Thammawichai, M. A double-layered hybrid approach for network intrusion detection system using combined naive bayes and SVM. *IEEE Access* **2021**, *9*, 138432–138450. [CrossRef]
34. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6.
35. Kumar, A.; Guleria, K.; Chauhan, R.; Upadhyay, D. Advancing Intrusion Detection with Machine Learning: Insights from the UNSW-NB15 Dataset. In Proceedings of the 2024 IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS), Bangalore, India, 28–29 June 2024; pp. 1–5.
36. Arun, C.B.; Anusha, M.; Rao, T.; Rohini, B.; Gargi, N.; Kodipalli, A. Enhancing Network Intrusion Detection using Artificial Neural Networks: An Analysis of the UNSW-NB15 Dataset. In Proceedings of the 2024 International Conference on Integrated Intelligence and Communication Systems (ICIICS), Kalaburagi, India, 22–23 November 2024; pp. 1–7.
37. Ullah, I.; Mahmoud, Q.H. A scheme for generating a dataset for anomalous activity detection in iot networks. In Proceedings of the Canadian Conference on Artificial Intelligence, Kingston, ON, Canada, 28–31 May 2020; pp. 508–520.
38. Elrawy, M.F.; Awad, A.I.; Hamed, H.F. Intrusion detection systems for IoT-based smart environments: A survey. *J. Cloud Comput.* **2018**, *7*, 1–20. [CrossRef]
39. Bebortta, S.; Senapati, D. Empirical characterization of network traffic for reliable communication in IoT devices. In *Security in Cyber—Physical Systems*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 67–90.
40. Whelan, J.; Sangarapillai, T.; Minawi, O.; Almehmadi, A.; El-Khatib, K. Uav attack dataset. *IEEE Dataport* **2020**, *167*, 1561–1573.
41. Choudhary, G.; Sharma, V.; You, I.; Yim, K.; Chen, R.; Cho, J.H. Intrusion detection systems for networked unmanned aerial vehicles: A survey. In Proceedings of the 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC), Limassol, Cyprus, 25–29 June 2018; pp. 560–565.
42. Krishna, C.L.; Murphy, R.R. A review on cybersecurity vulnerabilities for unmanned aerial vehicles. In Proceedings of the 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), Shanghai, China, 11–13 October 2017; pp. 194–199.
43. Attack, W.; Attack, I.; Attack, B.F. Ensemble of feature augmented convolutional neural network and deep autoencoder for efficient detection of network attacks. *Sci. Rep.* **2025**, *15*, 4267. [CrossRef] [PubMed]
44. Fu, Y.; Du, Y.; Cao, Z.; Li, Q.; Xiang, W. A deep learning model for network intrusion detection with imbalanced data. *Electronics* **2022**, *11*, 898. [CrossRef]
45. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; Volume 1.
46. Drewek-Ossowicka, A.; Pietrołaj, M.; Rumiński, J. A survey of neural networks usage for intrusion detection systems. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 497–514. [CrossRef]
47. Khalaf, L.I.; Alhamadani, B.; Ismael, O.A.; Radhi, A.A.; Ahmed, S.R.; Algburi, S. Deep learning-based anomaly detection in network traffic for cyber threat identification. In Proceedings of the Cognitive Models and Artificial Intelligence Conference, Prague, Czech Republic, 13–14 June 2024; pp. 303–309.
48. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
49. Jeong, H.J.; Lee, H.J.; Kim, G.N.; Choi, S.H. Self-MCKD: Enhancing the Effectiveness and Efficiency of Knowledge Transfer in Malware Classification. *Electronics* **2025**, *14*, 1077. [CrossRef]
50. Wu, Y.; Zang, Z.; Zou, X.; Luo, W.; Bai, N.; Xiang, Y.; Li, W.; Dong, W. Graph attention and Kolmogorov–Arnold network based smart grids intrusion detection. *Sci. Rep.* **2025**, *15*, 8648. [CrossRef] [PubMed]

51. Goldblum, M.; Fowl, L.; Feizi, S.; Goldstein, T. Adversarially robust distillation. In Proceedings of the AAAI conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 3996–4003.
52. Sheatsley, R.; Papernot, N.; Weisman, M.J.; Verma, G.; McDaniel, P. Adversarial examples for network intrusion detection systems. *J. Comput. Secur.* **2022**, *30*, 727–752. [CrossRef]