Pushing Network Forensic Readiness to the Edge: A Resource Constrained Artificial Intelligence Based Methodology

Syed Rizvi*, Mark Scanlon†, Jimmy McGibney*, John Sheppard*
*South East Technological University, Waterford, Ireland
Syed.Rizvi@postgrad.wit.ie, {Jimmy.McGibney, John.Sheppard}@setu.ie
†School of Computer Science, University College Dublin, Ireland
mark.scanlon@ucd.ie

***Abstract*—Rapid developments in recent years with the Internet of Things (IoT) have supported significant growth in edge computing. The growing number and diversity of IoT/edge devices increase the risk of security incidents. As many IoT/edge devices can be considered lightweight, with limited data processing capacity and significant heterogeneity, traditional digital forensic investigation techniques may not always work with them. Network forensic readiness on IoT/edge devices is a proactive approach to collecting evidence to assist with forensic examinations. This paper introduces the Network Forensic Readiness for Edge Devices (NetFoREdge) framework, focussing on deploying lightweight AI models in resource-constrained environments for attack detection, evidence collection, and preservation. The proposed lightweight AI-driven solution performed effectively on resource-constrained physical devices, namely a Raspberry Pi 3B and a Raspberry Pi Zero 2 W. To evaluate the effectiveness of this approach, experiments have been conducted using two datasets: the recently released IoT network attack dataset, CICIoT2023, and the IoT-23 dataset. The experimental results are very encouraging – achieving an accuracy rate exceeding 99.60% and 99.98% for multiclassification on CICIoT2023 and IoT-23 datasets, respectively, and demonstrating the feasibility of network forensic readiness on IoT/edge devices with limited memory, storage, CPU usage, and power consumption.**

***Index Terms*—Forensic Readiness, Internet of Things, Network Intrusion Detection Systems, Artificial Intelligence.**

I. Introduction

Increasingly complex network architectures and lower latency requirements have pushed computation and application services towards edge devices. Edge applications often have complex implementations, while edge devices present significant security threats [1]. These challenges are compounded by the increasing frequency of threats to IoT and edge networks that exploit devices and vulnerability of the protocol(s). A recent Red Hat report [2] highlights the challenge of security in edge development due to the increasing number of threats targeting edge infrastructure. In addition, both IoT forensics and forensic readiness are growing topics within the digital forensic research community [3]. Conventional digital forensic readiness approaches are not suitable for IoT devices due to their limited resources and lightweight nature [4].

IoT/Edge network forensics readiness can prove crucial to detecting and responding to security incidents [5]. IoT ecosystems, with their inherent vulnerabilities and lack of uniform standards, pose challenges to digital investigations. Finding the root cause can be facilitated by gathering evidence from various sources when anomalies arise. Network provenance plays a crucial role in attack detection and improving forensic investigations by providing evidence support. Although Network Intrusion Detection Systems (NIDS) are essential for network security in IoT environments, they can lack the ability to provide comprehensive post-attack forensic analysis.

Deep Learning (DL) has proven effective for network forensics [6] but its application on resource-constrained IoT/edge devices presents challenges. AI-powered solutions for network forensics require a significant amount of computational power, which consumes considerable resources and energy [7]. To address these challenges, this research presents an innovative solution: Expanding DL capabilities to resource-constrained devices for network forensic readiness. This approach reduces computational usage, power consumption, memory usage, and storage burden while maintaining precision and efficacy when detecting network attacks. In addition, it supports the collection and preservation of key evidence to improve network forensic analysis.

A. Contribution of This Work

The main contributions of this paper are as follows:

- Although other network forensic readiness frameworks/methodologies have been published in the literature, none have focused on resource-constrained environments. This paper outlines the design, implementation and evaluation of a novel framework for network forensic readiness in resource-constrained environments.
- The proposed framework uses less computational and power resources compared to existing approaches while achieving comparable results. In addition, it collects and preserves key evidence, helping investigators in post-attack network forensic analysis.
- A detailed analysis of nine different AI algorithms for multiclassification tasks.
- An analysis of resource utilisation of each of the proposed models and a demonstration of the viability of lightweight

AI models for network forensic readiness in resource-constrained environments.
- An evaluation of the framework using the CICIoT2023 and IoT-23 datasets using physically representative IoT/edge devices, namely a Raspberry Pi 3B and a Raspberry Pi Zero 2 W.

II. Related Work

In the field of IoT/edge networks, DL and Machine Learning (ML) techniques play a crucial role in improving network forensic readiness in resource-constrained scenarios [1, 6].

Recently, Wang et al. [8] introduced a lightweight Intrusion Detection System (IDS) model based on BIdirectional Long Short-Term Memory (DL-BiLSTM). This model effectively addressed the resource constraints commonly encountered in IoT devices. Using Incremental Principal Component Analysis (IPCA) and dynamic quantisation techniques, the model achieved good results in terms of both intrusion detection and computational efficiency on benchmark datasets, including CIC-IDS2017, N-BaIoT, and CICIoT2023. The results highlight the superiority of the DL-BiLSTM model over traditional DL models and the state-of-the-art in terms of detection accuracy while maintaining a lower model complexity.

Narayan et al. [9] introduced an ML-based framework that focusses on the latest CICIoT2023 dataset. Using the Random Forest (RF) algorithm, this framework achieved significant improvements in precision, recall, and F1 score when compared to existing methods. The framework exhibited accuracy rates of 99.68%, 99.44%, and 99.23% for binary classification, 8 distinct classes, and 34 distinct classes, respectively. The authors [10] proposed a lightweight DL-based IDS for binary classification, which addresses the concern of IDSs that require significant computational resources and processing time. They evaluated the proposed model using two datasets (CIC-IDS2017 and CSE-CIC-IDS2018), obtaining high accuracy of 99.7% and 99.98%, respectively. Danso et al. [11] proposed an innovative approach based on transductive transfer learning for the profile and identification of IoT devices, addressing device type identification, vulnerability assessment, and visualisation. This approach reflects robustness and validation against diverse datasets, confirming its suitability for IoT security.

In [12], the authors presented a multigranular attention Bi-LSTM-CRF model designed to extract Indicators of Compromise (IOCs) from a range of threat text sources. They employed a Heterogeneous Information Network (HIN) to contextualise the IOCs, manually defining meta-paths to illustrate relationships among them. This study focused on six key categories: attacker, vulnerability, device, platform, malicious file, and attack type. Although the precision of IOC extraction reached 99.86%, it varied across different items. Furthermore, the multigranular model achieved a precision of 98.72% in threat entity recognition compared to other methods tested.

In the landscape of digital forensics for smart environments, Babun et al. [13] presented IoTDots, a novel framework composed of a modifier and an analyser. The approach combined source code review and data processing to extract valuable forensic information related to device activity during incidents. Meffert et al. [14] proposed a model to acquire forensic state information from IoT devices through a centralised controller. The model detected alterations in the infected devices for the construction of a chronological timeline. However, it is essential to note the challenge of retrieving deleted or historical data, which can hinder comprehensive investigative efforts.

A survey on deep neural networks was conducted that reduces the processing, storage, and energy requirements of IoT applications by Mishra and Gupta [15]. The authors divided the strategies into five categories: knowledge distillation, bit precision, network pruning, sparse representation, and miscellaneous. The authors examined each category in detail and identified their current challenges.

In the state of the art, there has been significant progress in developing a wide range of techniques for IDS for IoT networks/resource-constrained environments. However, despite these advances, the critical area of evidence collection in such environments remains largely unexplored. To address this research gap, this study proposes an AI-based framework for resource-constrained environments. The framework focusses on evidence collection after detecting various types of network attacks, facilitating investigators in conducting post-attack network forensic analysis.

III. Methodology for Network Forensic Readiness for Edge Devices

The proposed Network Forensic Readiness for Edge Devices (NetFoREdge) framework uses an AI-based model designed to operate efficiently within resource-constrained environments. NetFoREdge continuously monitors real-time network traffic to differentiate between normal network activity and potentially malicious behaviour. Once an attack is identified, NetFoREdge employs AI models to accurately categorise the attack in progress and begins collecting evidence and saving it. This process unfolds systematically, encompassing the acquisition and archiving of evidence metadata associated with the detected malicious activity. The evidence collection includes diverse data, such as network traffic logs, packet captures, and other relevant data points. NetFoREdge avoids immediate intervention or countermeasures against the attack, instead focussing on the preservation of evidence. The objective is to ensure the safety of digital evidence until it is required by forensic analysts for further investigation. Figure 1 illustrates the framework of the proposed work for IoT/edge devices.

While traditional NIDS may struggle with the limitations imposed by constrained computational, power, memory, and storage resources, NetFoREdge is optimised for efficiency. Its operation is based on the use of lightweight AI models, which minimise computational, memory, power, and storage concerns while maintaining a high level of accuracy.

In a resource-constrained environment, the ability to identify, categorise and preserve data relevant to possible security incidents is developed as an essential element of maintaining

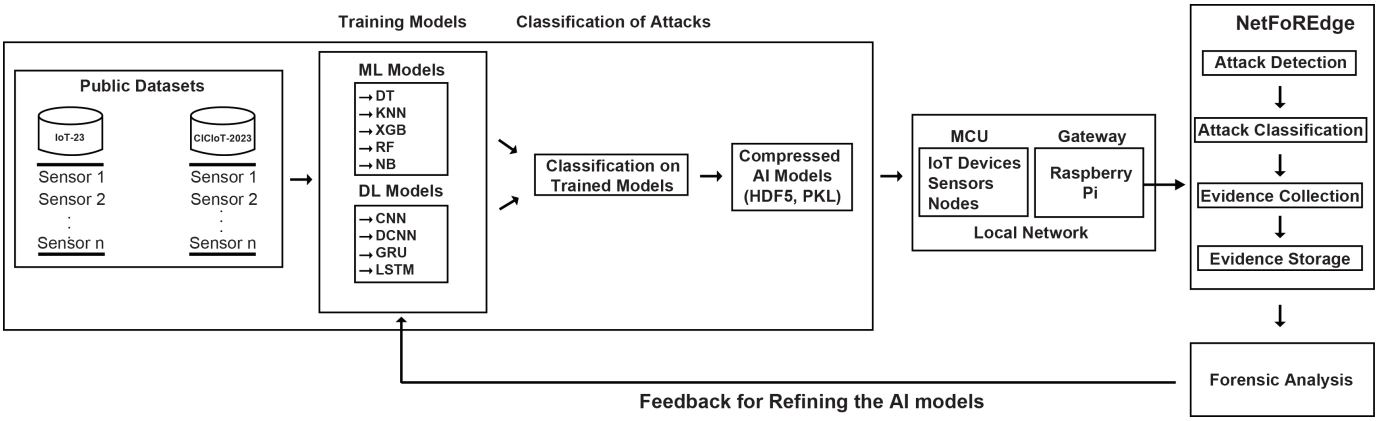

Fig. 1: The Framework of the Proposed IoT/Edge Forensic Readiness Methodology

network security. NetFoREdge provides forensic analysts with the tools and data they need to conduct investigations by precisely recognising and categorising network attacks and systematically collecting digital evidence.

A. Experimental Setup

The experimental setup was chosen to facilitate the creation and modification of ML and DL techniques for resource-constrained environments. The primary objective was to explore diverse model architectures and hyperparameters to identify optimal configurations that deliver exceptional performance while minimising computational, memory, power, and storage demands. To achieve this, systematic changes were made to components such as layers, hyperparameters, activation functions, kernel sizes, and loss functions in DL models, along with dilation rates in 1D-DCNN. The aim was to investigate these components to achieve the optimum balance between model accuracy and computational efficiency. The slack environment and physical environments used as part of this work are outlined in the following subsections.

1) Slack Environment: The slack environment served as a platform for data pre-processing and the initial development and testing of the proposed models. The slack environment consisted of hardware that included an Intel Core i7-1165G7 processor with 32GB of RAM. The model development was carried out using Python version 3.9.7, with `scikit-learn` for ML models, and `TensorFlow` and `Keras` for DL models. Data preprocessing tasks were efficiently handled using the `Pandas` and `NumPy` libraries. The `Seaborn` library was used for data visualisation and analysis.

2) Resource-constrained Environment: The importance of establishing an environment that accurately simulates the limitations caused by limited resources was achieved through the utilisation of multiple physical devices, including a Raspberry Pi 3B and Pi Zero 2 W (both having a 64-bit architecture). The Raspberry Pi is popular in the literature on resource-constrained environments [16, 17, 18]. The use of these devices highlights the feasibility and significance of research in the context of IoT/edge security. The selection of Raspberry Pi 3B and Pi Zero 2 W over more powerful in terms of resources, newer models such as the Raspberry Pi 4 or Pi 5 demonstrate the applicability of the proposed model to low-end devices without impacting the primary task. It provides a real-world scenario for the performance evaluation of the proposed models in resource-constrained environments. As a result, reliable and useful insights into the effectiveness of the proposed models in realistic resource-constrained environments were achieved. The details of the software and hardware used for experiments are summarised in Table I.

B. Dataset Description

The first dataset used in this work, the CICIoT2023 [19] dataset, is a significant recent contribution to IoT security research created by the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick – providing a new benchmark for large-scale attacks in IoT environments. It includes an IoT topology with 105 devices and 33 attacks categorised into seven classes: Denial of Service (DDoS), Distributed DoS (DDoS), Recon, Web-based, Brute Force, Spoofing, and Mirai. This dataset enables researchers to evaluate security analytics applications and advance the field. A notable strength of this dataset is the inclusion of realistic IoT attack scenarios. It provides comprehensive documentation and data collection procedures for each attack. Based on the popularity of previously released CIC datasets, combined with the size of the dataset and the variety of attacks present, CICIoT2023 is likely to become a popular benchmarking dataset to evaluate ML and DL algorithms to detect malicious IoT network traffic.

The second dataset used in this work is the IoT-23 dataset created by the Avast AIC laboratory [20]. The dataset is derived from IoT network traffic and contains communication data from three benign IoT devices and twenty malware scenarios. There are 325,307,990 records in total, with 294,449,255 being malicious records. There are 21 features in the dataset.

C. Pre-processing on Datasets

Exploratory Data Analysis (EDA) plays a crucial role in optimising the performance of AI models. In this paper, EDA

	Slack Environment	Resource-constrained Environment	
Devices	Dell Desktop	Raspberry Pi 3B	Raspberry Pi-Zero 2 W
Hardware Specification	Processor: Core i5, RAM: 32GB, Storage: 512GB	Processor: Quad Core (1.2GHz), RAM: 1GB, Storage: 16GB	Processor: Quad Core (1.0 GHz), RAM: 512MB, Storage: 8GB
Processor Architecture	64-bit	64-bit	64-bit
Operating System	Windows 10	Linux Debian V:12	Linux Debian V:12
Connectivity	Ethernet	Wifi	Wifi
Software Libraries	Tensorflow, Keras, Scikit-learn, Numpy, Pandas, Seaborn	Tensorflow, Keras, Scikit-learn, Numpy, Pandas, Seaborn	Tensorflow, Keras, Scikit-learn, Seaborn
IDE	Jupyter Notebook	Geany	Geany
Programming Language	Python: 3.9.7	Python: 3.11.6	Python: 3.11.6

TABLE I: Apparatus Table

was performed on a Comma Separated Values (CSV) file of the CICIoT2023 dataset and the IoT-23 data frames. The data preprocessing involved eliminating duplicate entries, removing null and missing values, addressing inconsistent data, and discarding irrelevant features. The revised attack distribution showed a notable decrease in instances for each class.

To address class imbalance, ensemble techniques were employed to rectify the disparate representation of attack classes. These methods improve predictive performance by combining multiple classifiers or models, each trained on various subsets of the imbalanced data. The refined distribution across the attack classes on the CICIoT2023 dataset was as follows: DDoS (1,941,636), DoS (467,412), Mirai (151,934), Benign (63,767), Spoofing (28,322), Recon (20,536), Web-based (1,410), and Brute Force (749). To address class imbalance in the IoT-23 dataset, attacks C&C-Mirai, C&C-Torii, and FileDownload were excluded due to fewer instances.

A preprocessing label encoder was applied to convert the attack class labels into a numeric representation. The label encoder assigned a unique numerical value to each attack class, transforming them from categorical variables into numerical labels. This conversion facilitated the use of various ML algorithms and techniques that require numeric inputs. The datasets were partitioned into training and test sets in an 80:20 and 70:30 ratio for CICIoT2023 and IoT-23 respectively, to enable effective model evaluation and validation.

D. Deep Learning Algorithms

1) Convolutional Neural Network: 1D Convolutional Neural Networks (1D-CNNs) were considered due to cost-effective and time-saving real-time implementation using 1D convolutions, making these networks suitable for real-time attack detection and monitoring. The 1D-CNN model has been optimised for feature extraction and classification. Starting with 32 filters-based convolutional layers arranged to catch localised patterns, this procedure yields a tensor shape (None, 10, 32) with variable batch sizes (None), a sequence length of 10, and 32 filters; acting as an effective feature extractor capable of extracting critical features from datasets.

Subsequently, the max-pooling layer intelligently reduces dimensionality while preserving essential features. This process eliminates less important features from consideration and selects a shape of (None, 5, 32) for the output, enhancing computing efficiency. Next, a convolutional layer with 64 filters is incorporated to further expand the architecture, allowing the extraction of complex features from the data to create an effective representation. This layer shortens the sequence length to 3, and as the number of filters increases, it generates an output shape of (None, 3, 64). This improves the classification capabilities of the model.

Following the layers within the architecture, including a flattening layer to convert the output of the convolutional layer to one-dimensional vectors, a dense layer has been added to classify the output by utilising the `softmax` activation function. All layers process and transform extracted features for more accurate classification.

2) Dilated Convolutional Neural Network: The 1D Dilated Convolutional Neural Networks (1D-DCNNs) offer distinct advantages over traditional CNNs [10]. This paper introduces an optimised 1D-DCNN model that demonstrates strong feature extraction and classification capabilities. The hyperparameter fine-tuning plays a crucial role in handling complex, real-world challenges. The proposed model was initiated with a convolutional layer of 32 filters, a kernel size of 1, and a dilation rate of 2, which efficiently extracted patterns and essential attributes from the data. This dilated convolution layer expanded the contextual scope for information extraction. This layer served as a repository of learnt representations upon which subsequent layers were constructed. The model was then expanded by adding another convolutional layer with 64 filters, 1 kernel size, and 8 dilation rates to further extract complex details and more complex representations through the additional layer. A higher dilation rate helps capture complex relationships and insightful patterns with more precision. Both layers act as efficient feature extraction mechanisms.

Flattening layers reshape the output to a one-dimensional format to ensure compatibility with dense layers, enabling efficient integration. The final dense layer uses `softmax` activation for multi-class predictions, with weights optimised by the `Adam` algorithm.

E. Recurrent Neural Network

1) Long Short-Term Memory: Recurrent Neural Networks (RNNs) are known to be computationally intensive due to the backpropagation-through-time process [21]. To address this

challenge, the Long Short-Term Memory (LSTM) architecture was introduced as an RNN variant, optimised for training efficiency over longer time intervals. The model architecture includes an LSTM layer and a dense layer, designed to capture and learn relationships over time.

Traditional RNNs often experience the vanishing gradient problem, which LSTM layers avoid by employing their gating mechanism. The output shapes (None, 64) and 19,712 trainable parameters allow this model to capture long-term dependencies while maintaining contextual memory – making it particularly effective when dealing with sequences characterised by temporal gaps. Following the LSTM layer, a dense layer with 650 trainable parameters is added, providing a deeper abstraction of the learnt features. This structure yields simple, interpretable representations, offering valuable insights into reasoning processes within models.

2) Gated Recurrent Unit: Gated Recurrent Unit (GRU) models are an ideal solution for real-time applications as they require fewer training parameters and therefore consume less memory. The proposed model contains 14,976 trainable parameters. GRU layers recognise long-term dependencies and bypass the limitations of traditional RNNs through their gating mechanisms. These gates manage the flow of information by selectively updating and resetting the hidden state, enabling adaptive retention or the discarding of information from previous time steps. The GRU layer efficiently learns temporal dependencies, making it suitable for analysing sequences with variable lengths and complex patterns in applications like natural language processing and sentiment analysis.

A dense layer is built upon a GRU layer with nonlinear transformation to improve the model's performance. Its output shape is (None, 10), with 650 parameters. This layer converts GRU representations into class probabilities for prediction across multiple categories and enhances decision-making and interpretability by providing insights into learnt features.

F. Machine Learning Algorithms

An experimental approach was used to ensure comprehensive and impartial model selection. This technique involved training and evaluating well-established ML algorithms such as K-Nearest Neighbours (KNN), Random Forest (RF), Decision Trees (DT), XGBoost (XGB), and Naïve Bayes (NB). The `GridSearchCV` function was used to maximise the predictive accuracy and robust generalisation of the performance of each model. Hyperparameter optimisation involves creating a parameter grid that contains relevant hyperparameter values specific to each model. For example, in the case of the RF model, parameters such as the number of estimators, the maximum features per split, and the splitting criteria were systematically taken into account. The `GridSearchCV` algorithm evaluated various combinations of hyperparameters available within each model's application space before selecting the optimal set for each one. Following the hyperparameter optimisation phase, the models were trained on the datasets.

IV. Results and Findings

The optimal performance in resource-constrained environments was achieved by tuning the hyperparameters of the model to not consume more than 60% of memory for any model. To evaluate CPU usage, the `psutil` library was used.

A. Results Analysis

With the CICIoT2023 dataset, RF and DT achieved accuracy rates of 99.58% and 99.47% respectively; while XGB reached 99.60%. To optimise DL models on this dataset, 15 training epochs were reduced to 6 using an early stopping technique to limit overfitting, leading to accuracy rates between 98.80% and 99.06%, showing their effectiveness in capturing related patterns and relationships.

In terms of training time on both datasets, NB required less training time; however, its accuracy is not sufficient compared to other models. KNN notably required fewer training times compared to other proposed models, recording `162.9s` for CICIoT2023 and `0.5s` for IoT-23. However, its validation time was higher due to its lazy learning nature, which involves distance calculations between data points during the prediction. DT required less training time on both PC and Raspberry Pi, with significant accuracy. The RNN models, i.e., LSTM and GRU, required less training time in resource-constrained environments compared to CNN models. DCNN outperformed traditional CNN in terms of training time; however, it achieved slightly lower accuracy. Tables II and III provide a detailed examination of the characteristics of each model on the CICIoT2023 and IoT-23 datasets.

SNo	Model	PC Training Time (s)	PC Validation Time (s)	PC CPU Usage (%)	Pi 3B Training Time (s)	Pi 3B Validation Time (s)	Pi 3B CPU Usage (%)	Pi Zero 2 W Validation Time (s)	Pi Zero 2 W CPU Usage (%)	Accuracy (%)	F1 Score
1	DT	24.16	0.12	0.21	397.38	1.31	21.80	26.04	28.75	99.47	0.994
2	RF	484.35	5.91	12.43	5,331.78	133.32	26.50	84.24	34.42	99.58	0.995
3	KNN	0.23	35,143.31	15.56	162.98	386,573.65	31.24	567,525.42	52.91	99.19	0.992
4	NB	1.37	1.12	4.01	648.47	29.17	19.10	450.98	24.53	78.40	0.712
5	XGB	687.39	0.84	41.27	1,611.57	12.31	44.38	48.18	42.10	99.60	0.994
6	1D-CNN	1,160.30	313.56	35.25	27,004.48	340.81	39.75	368.81	54.45	98.80	0.988
7	1D-DCNN	1,146.76	182.83	34.43	11,689.77	344.44	37.85	337.33	47.20	98.92	0.987
8	LSTM	396.19	198.23	19.50	8,622.88	264.79	26.50	359.07	43.23	99.06	0.989
9	GRU	376.21	198.93	19.30	7,893.12	251.17	26.00	337.41	41.22	99.01	0.989

TABLE II: Evaluation on the CICIoT2023 Dataset

With the IoT-23 dataset, the ML models outperformed the DL models; in particular, XGB reached 99.98% accuracy, while DT and RF achieved 99.94% and 99.89%, respectively. DL models (LSTM, GRU, 1D-CNN, and 1D-DCNN) achieved accuracies between 97.85 and 99.16 %, showing their ability to identify complex patterns.

A comprehensive comparison was carried out between the proposed DL models and those presented by Wang et al. [8]. This analysis included an examination of the size of the models, the performance results, and the efficiency of the

SNo	Model	PC Training Time (s)	PC Validation Time (s)	PC CPU Usage (%)	Pi 3B Training Time (s)	Pi 3B Validation Time (s)	Pi 3B CPU Usage (%)	Pi Zero 2 W Validation Time (s)	Pi Zero 2 W CPU Usage (%)	Accuracy (%)	F1 Score
1	DT	0.27	2.00	1.54	1.12	2.02	28.62	2.49	30.40	99.94	0.999
2	RF	8.93	2.26	2.41	34.89	3.47	25.61	3.27	40.30	99.89	0.999
3	KNN	0.16	3.16	0.96	0.56	14.74	25.44	15.89	28.80	90.94	0.912
4	NB	0.02	2.02	0.40	0.12	2.16	27.52	2.43	53.20	58.85	0.458
5	XGB	1.95	2.03	7.61	13.35	2.22	77.61	2.37	95.53	99.98	0.999
6	1D-CNN	58.69	2.76	1.41	749.55	8.91	10.43	16.51	13.20	99.16	0.991
7	1D-DCNN	67.62	2.62	1.01	628.32	6.89	8.21	15.20	12.35	97.85	0.979
8	LSTM	70.74	2.72	1.42	509.74	10.65	11.05	18.10	11.27	97.97	0.981
9	GRU	75.62	2.66	1.22	547.42	14.05	9.89	17.70	11.39	98.45	0.982

TABLE III: Evaluation on the IoT-23 Dataset

Model	Training (s)	Accuracy (%)	Size (KB)	Parameters
CNN [8]	1,515.4	92.2	184.1	42,952
Proposed CNN	1,160.3	98.8	298.7	32,330
LSTM [8]	764.8	92.7	88.4	21,128
Proposed LSTM	396.2	99.0	350.2	20,362
DL-BiLSTM [8] (Best performed)	708.4	93.1	36.5	1,988
Proposed LSTM (Best performed)	396.2	99.0	350.2	20,362

TABLE IV: DL Models Compared with [8] on CICIoT2023

training, as presented in Table IV. The results clearly demonstrate that the proposed models possess clear advantages, with lighter models simultaneously attaining higher accuracy with reduced computational demands and computational costs. The PowerTop utility was used to monitor the energy consumption of each AI model using a default 20 second interval setting. Table V presents the total power consumption of each AI model. The resource-constrained experiments and the corresponding results illustrate that both the DL and ML models have significant potential for NetFoREdge.

AI Model	CPU Train (mW)	AI Model Train (mW)	CPU Test (mW)	AI Model Test (mW)
DT	5,490	876	5,030	327
RF	5,670	1,470	5,750	367
KNN	4,820	911	5,600	1,510
NB	4,580	416	3,850	279
XGB	6,780	3,260	4,980	335
1D-CNN	5,600	1,700	4,510	881
1D-DCNN	5,400	1,550	4,360	605
LSTM	6,000	2,020	4,950	871
GRU	5,840	1,630	4,250	635

TABLE V: CPU & AI Models Power Consumption During Training and Testing Phase on a Raspberry Pi 3B

V. Comparison And Discussion

The proposed approaches performed efficiently by achieving higher accuracy in resource-constrained environments while consuming less power and requiring less training time compared to existing approaches, e.g., the method proposed by ElSayed et al. [22] required 7,500mW on the Raspberry Pi 4B. Agbedanu et al. [23] utilised a Raspberry Pi 4 with 2GB RAM and 32GB storage, along with an AMD V7 quad-core processor, and achieved a power consumption of 259mW. Their proposed model required 149s to train the model. Similarly, the approach proposed by Bhandari et al. [24] employed a Raspberry Pi 4 with 8GB RAM and an ARM Cortex-A72 processor, consuming significantly less power (237mW).

Although existing approaches demonstrate lower power consumption compared to this work, it is essential to note that the proposed method is deployed on a cost-effective Raspberry Pi 3B with 1GB RAM, 16GB storage, and a Cortex-A53 (ARMv8) processor. Despite this, the proposed method requires only a negligible increase in power consumption, i.e., 327mW and 371mW for IoT-23 and CICIoT2023 datasets, respectively. However, it significantly outperforms existing techniques in terms of accuracy and training time, achieving an impressive accuracy of 99.6% and 99.9% for CICIoT2023 and IoT-23 respectively. The proposed model required `397.3s` and `1.1s` of training time for the CICIoT2023 and IoT-23 datasets, respectively, as shown in Table VI.

Ref.	Device	Dataset	Accuracy (%)	Training (s)	Power (mW)
Agbedanu et al. [23]	RPi 4B	UNSW-NB15	97.6	149.0	259
Bhandari et al. [24]	RPi 4B	IoT-23/EdgeIIoTset	95.0/96.0	-	237
ElSayed et al. [22]	RPi 4B	CICIoT2023	93.6	-	7,500
Proposed Model	**RPi 3B**	**IoT-23**	**99.9**	**1.1**	**327**
Proposed Model	**RPi 3B**	**CICIoT2023**	**99.6**	**397.3**	**371**

TABLE VI: Power Comparison of Proposed Work with Existing Approaches

As shown in Table VII, despite the resource-constrained limitation of the presented work, the proposed models achieved results comparable to existing research (notably with those existing approaches having more powerful systems) and also achieved better results than existing work using higher-end, nonresource-constrained systems.

Reference	Dataset	Classification	Low-end device	Accuracy (%)
Nguyen et al. [25]	IoT-23	Multi-class	✓	73.0
Rizvi et al. [1]	IoT-23	Multi-class	✓	99.9
Proposed model	**IoT-23**	**Multi-class**	✓	**99.9**
ElSayed et al. [22]	CICIoT2023	Binary	✓	93.6
Narayan et al. [9]	CICIoT2023	Multi-class	✗	99.4
Le et al. [26]	CICIoT2023	Multi-class	✗	99.5
Proposed model	**CICIoT2023**	**Multi-class**	✓	**99.6**

TABLE VII: Accuracy Comparison of Proposed Models with Existing Approaches

The results demonstrate that the selection of an AI model for a resource-constrained environment depends on factors

such as the nature of the data, sample size, and intended application. Customising the AI model to address the unique aspects of the problem is crucial. In particular, training AI models in resource-constrained environments offers distinct benefits. For example, retraining models with specific traffic patterns on individual devices can enhance evidence collection. Furthermore, this method is advantageous in remote areas with limited or no Internet connectivity and ensures privacy and data security by avoiding data transfer to the cloud.

Power consumption is a crucial factor when selecting an AI-driven solution for resource-constrained environments. The power requirements for the AI model training phase depend on factors such as the size of the training dataset. This study utilised the Running Average Power Limit (RAPL), a CPU hardware feature, to measure each AI model's power consumption. Although RAPL is slower compared to hardware, communication with the power engine should be examined to prevent undesirable behaviours. The power metre may detect minor variations in power consumption.

A. Benefits

Nik Zulkipli and Wills [27] summarises the general advantages of forensic readiness for IoT devices, which apply equally to the work presented as part of this paper. These include preparing for the future needs for digital evidence, reducing digital investigation costs, preventing insider bad actors from covering their tracks, improving corporate governance, and reducing the costs of statutory disclosure requirements. NetFoREdge contributes the following additional benefits in edge forensics, specifically for resource-constrained devices:

- NetFoREdge utilises fewer resources in terms of CPU load and power, which enables localised network forensic readiness in resource-constrained environments.
- NetFoREdge can seamlessly be deployed on physical IoT and Edge devices with negligible impact on operational efficiency.
- NetFoREdge allows for the prioritisation of pertinent local edge-based data collection, preservation, and analysis based on each attack's potential severity and impact.

B. Limitations and Future Work

Although the approach outlined as part of this paper has several benefits, a number of limitations have also been identified, each of which is addressable in future work:

- The requirement to use the device's CPU and memory for the training and/or testing of the proposed approach necessitates that a suitable amount of free CPU cycles and memory be available on the already resource-constrained device, which already has its existing task(s) to perform.
- Although the proposed models exhibit promise within controlled environments, their practical deployment in diverse real-world scenarios has not been extensively examined. Factors such as varying network conditions, device heterogeneity, and environmental disturbances may pose challenges that impact the models' performance and reliability in real-world deployments.

In the future, the objective is to evaluate a system by monitoring live network traffic and analysing various parameters such as throughput, latency, and memory usage. The aim is to determine how resource-constrained devices are affected and whether these impacts compromise the routine functioning of IoT and edge devices. Future directions worth exploring include deploying lightweight AI-based models across a variety of IoT and edge devices to check their scalability and performance across diverse configurations.

VI. Conclusion

To enhance security and aid forensic readiness in complex, heterogeneous networks, this paper proposes training and deploying lightweight network traffic classification algorithms directly on low-powered physical Raspberry Pi devices to increase protection, while categorising pertinent traffic for storage and potential further investigation. AI models have demonstrated real-time detection of malicious attacks with minimal computational, memory, power, and storage use, making them suitable for resource-constrained environments. The models were validated using the CICIoT2023 and IoT-23 datasets, achieving impressive accuracy rates of 99.60% and 99.99%, respectively, across several distinct attack types with minimal resource usage, while also attaining F1 scores of 99.44% for CICIoT2023 and 99.97% for IoT-23. These findings provide valuable information on training time, validation time, CPU usage, and power consumption when selecting lightweight algorithms for forensic readiness. These results highlight the ability of the proposed models to prepare systems by facilitating the detection, categorisation, and systematic preservation of evidence for subsequent forensic analysis, significantly advancing readiness for subsequent forensic analysis.

References

[1] S. Rizvi, M. Scanlon, J. McGibney, and J. Sheppard, "An Evaluation of AI-Based Network Intrusion Detection in Resource-Constrained Environments," in *2023 IEEE 14th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2023, pp. 0275–0282.

[2] E. Hanselman, D. Immerman, J. Lyman, "The State of Edge Security Report," March 2024. [Online]. Available: https://www.redhat.com/en/blog/state-edge-security-report

[3] F. Breitinger, J.-N. Hilgert, C. Hargreaves, J. Sheppard, R. Overdorf, and M. Scanlon, "DFRWS EU 10-Year Review and Future Directions in Digital Forensic Research," *Forensic Science International: Digital Investigation*, vol. 48, p. 301685, 03 2024.

[4] F. I. Fagbola and H. S. Venter, "Smart Digital Forensic Readiness Model for Shadow IoT Devices," *Applied Sciences*, vol. 12, no. 2, 2022.

[5] N. H. Nik Zulkipli and G. B. Wills, "An Exploratory Study on Readiness Framework in IoT Forensics," *Procedia Computer Science*, vol. 179, pp. 966–973, 2021,

5th International Conference on Computer Science and Computational Intelligence 2020.

[6] S. Rizvi, M. Scanlon, J. McGibney, and J. Sheppard, "Application of Artificial Intelligence to Network Forensics: Survey, Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 110 362–110 384, 2022.

[7] N. Tekin, A. Acar, A. Aris, A. S. Uluagac, and V. C. Gungor, "Energy Consumption of On-Device Machine Learning Models for IoT Intrusion Detection," *Internet of Things*, vol. 21, p. 100670, 2023.

[8] Z. Wang, H. Chen, S. Yang, X. Luo, D. Li, and J. Wang, "A lightweight Intrusion Detection Method for IoT Based on Deep Learning and Dynamic Quantization," *PeerJ Computer Science*, vol. 9, p. e1569, 2023.

[9] K. R. Narayan, S. Mookherji, V. Odelu, R. Prasath, A. C. Turlapaty, and A. K. Das, "IIDS: Design of Intelligent Intrusion Detection System for Internet-of-Things Applications," *arXiv*, vol. abs/2308.00943, 2023.

[10] S. Rizvi, M. Scanlon, J. McGibney, and J. Sheppard, "Deep learning based network intrusion detection system for resource-constrained environments," in *Digital Forensics and Cyber Crime*, S. Goel, P. Gladyshev, A. Nikolay, G. Markowsky, and D. Johnson, Eds. Cham: Springer Nature Switzerland, 2023, pp. 355–367.

[11] P. K. Danso, S. Dadkhah, E. C. P. Neto, A. Zohourian, H. Molyneaux, R. Lu, and A. A. Ghorbani, "Transferability of Machine Learning Algorithm for IoT Device Profiling and Identification," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[12] J. Zhao, Q. Yan, X. Liu, B. Li, and G. Zuo, "Cyber threat intelligence modeling based on heterogeneous graph convolutional network," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 241–256.

[13] L. Babun, A. K. Sikder, A. Acar, and A. S. Uluagac, "IoTDots: A Digital Forensics Framework for Smart Environments," *arXiv*, vol. abs/1809.00745, 2018.

[14] C. Meffert, D. Clark, I. Baggili, and F. Breitinger, "Forensic State Acquisition from Internet of Things (FSAIoT): A General Framework and Practical Approach for IoT Forensics through IoT Device State Acquisition," in *Proceedings of the 12th International Conference on Availability, Reliability and Security*, ser. ARES '17. New York, NY, USA: Association for Computing Machinery, 2017.

[15] R. Mishra and H. Gupta, "Transforming Large-Size to Lightweight Deep Neural Networks for IoT Applications," *ACM Comput. Surv.*, vol. 55, no. 11, feb 2023.

[16] X. Liu, Q. Zeng, X. Du, S. L. Valluru, C. Fu, X. Fu, and B. Luo, "SniffMislead: Non-Intrusive Privacy Protection against Wireless Packet Sniffers in Smart Homes," in *Proceedings of the 24th International Symposium on Research in Attacks, Intrusions and Defenses*, ser. RAID '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 33–47.

[17] O. Gómez-Carmona, D. Casado-Mansilla, D. López-de Ipiña, and J. García-Zubia, "Optimizing computational resources for edge intelligence through model cascade strategies," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7404–7417, 2022.

[18] L. Ardito and M. Torchiano, "Creating and evaluating a software power model for linux single board computers," in *Proceedings of the 6th International Workshop on Green and Sustainable Software*, ser. GREENS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–8.

[19] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," *Sensors*, vol. 23, no. 13, 2023.

[20] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A Labeled Dataset With Malicious and Benign IoT Network Traffic," Jan. 2020, https://www.stratosphereips.org/datasets-iot23.

[21] T.-T.-H. Le, J. Kim, and H. Kim, "An effective intrusion detection classifier using long short-term memory with gradient descent optimization," in *2017 International Conference on Platform Technology and Service (PlatCon)*, 2017, pp. 1–6.

[22] Z. ElSayed, N. Elsayed, and S. Bay, "A Novel Zero-Trust Machine Learning Green Architecture for Healthcare IoT Cybersecurity: Review, Analysis, and Implementation," *arXiv*, vol. abs/2401.07368, 2024.

[23] P. R. Agbedanu, R. Musabe, I. Gatare, J. Rwigema, and Y. Pavlidis, "Exploring a Hybrid Technique to Detect Network Intrusions in IoT Systems using PCC, ADASYN, and ALMA," *TechRxiv*, 02 2023. [Online]. Available: http://dx.doi.org/10.36227/techrxiv.21948608.v1

[24] G. Bhandari, A. Lyth, A. Shalaginov, and T. M. Grønli, "Distributed Deep Neural-Network-Based Middleware for Cyber-Attacks Detection in Smart IoT Ecosystem: A Novel Framework and Performance Evaluation Approach," *Electronics*, vol. 12, no. 2, 2023.

[25] T. B. Nguyen, D. D. K. Nguyen, B. N. Le Nguyen, and T. Le, "A machine learning-based anomaly packets detection for smart home," in *Proceedings of the 12th International Symposium on Information and Communication Technology*, ser. SOICT '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 816–823.

[26] T. T. H. Le, R. W. Wardhani, D. S. C. Putranto, U. Jo, and H. Kim, "Toward Enhanced Attack Detection and Explanation in Intrusion Detection System-Based IoT Environment Data," *IEEE Access*, vol. 11, pp. 131 661–131 676, 2023.

[27] N. H. Nik Zulkipli and G. B. Wills, "An Exploratory Study on Readiness Framework in IoT Forensics," *Procedia Computer Science*, vol. 179, pp. 966–973, 2021, 5th International Conference on Computer Science and Computational Intelligence 2020.