

Classifying Internet of Things (IoT) devices and non-IoT devices in data center edge network using statistical and network flow features by one dimensional convolutional neural network

1st Sharath Ramamurthy

Upgrad Education Pvt. Ltd., Mumbai, India
Liverpool John Moores University, London, United Kingdom
sharu.murthy@gmail.com

2nd Dipankar Dutta

CSE Department, University Institute of Technology
The University of Burdwan, Burdwan, India
dipankar2222@yahoo.com

Abstract—The development of edge computing has led to the aggregation of traffic from IoT and non-IoT devices at edge data center networks located closer to cloud and computing resources. To provide application-specific Quality of Service (QoS) for these devices, classification of IoT and non-IoT devices at edge data centers is crucial. Our research suggests using a simple, compact, and computationally less expensive shallow one-dimensional convolutional neural network (1D-CNN) for real-time processing needs. The model was trained on a dataset consisting of 22 IoT devices and 7 non-IoT devices. The network flow and statistical features extracted from the network traces of these devices were fed into the model. The research demonstrated that with just 15 features, the model can achieve around 94% accuracy, and with 55 features, nearly 97% accuracy in classifying IoT and non-IoT devices.

Index Terms—IoT, Classification, 1D-CNN, Networking, Data Center, Edge Computing

I. INTRODUCTION

One of the newest developments in IoT is the introduction of edge computing, which brings computation and cloud resources closer to the data source (IoT devices). This crucial criterion makes it possible to handle data from time-critical IoT devices (such as data from a driverless car's live camera) with the least amount of latency. The edge data centers, which are situated closer to IoT devices, are one of the key elements of the edge computing idea. Data traffic from numerous IoT and non-IoT devices is frequently aggregated by edge data center switches and routers. For additional data processing, the aggregated traffic is sent to computation and storage units (servers) over the data center network. Typical IoT and non-IoT devices connected to an edge data center are shown in Fig. 1.

In the early days of the Internet, network traffic classification was one of the most popular research areas. Recently, this interest has expanded to include the IoT. For the classification of network traffic in the past, deep packet inspection (DPI) and port-based approaches were used [1]. The Internet Assigned Numbers Authority (IANA) assignments of application port numbers, which are used in protocols like Internet Protocol

(IP), User Datagram Protocol (UDP), and Transport Control Protocol (TCP), formed the basis of port-based approaches. However, due to the lack of standardized port numbers for all planned apps, port-based techniques proved unreliable. Later, DPI-based techniques were used to peer further into packets to identify the program, but they required a lot of computing power and were also useless when network traffic payloads were encrypted. Machine learning (ML) and deep learning (DL) approaches have recently gained popularity and have demonstrated outstanding accuracy in the classification of IoT

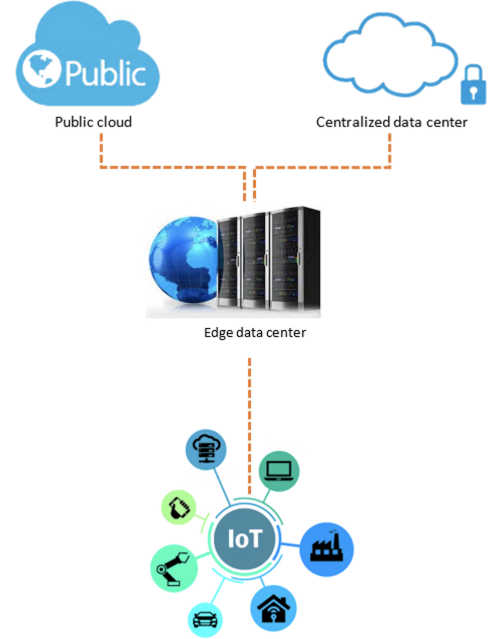


Fig. 1. Edge data center connecting IoT and non-IoT devices

devices. IoT traffic exhibits a consistent pattern and predictable behavior, which can be used to identify and categorize it using ML models. Traffic aggregators such as edge data center switches may receive IoT-related traffic (both from IoT and non-IoT devices) [2], [3]. IoT device traffic, however, cannot be handled in the same manner as traffic from non-IoT devices. Depending on the business use cases, there could be different QoS (bandwidth, latency, etc.) needs, security requirements, and requirements for policy enforcement. To be effective and deployable, the classification framework requires that edge data center switches be able to classify network traffic from IoT and non-IoT devices in real time.

In the beginning, CNNs were primarily created to solve picture categorization problems [4]. Later, they were applied to a wide range of DL issues, including time series forecasting [5], spatio-temporal models [6], and other problems. CNNs can be divided into one-dimensional (1D), two-dimensional (2D), and three-dimensional (3D) architectures depending on the type of input (1D/2D/3D). CNNs with more than three dimensions are uncommon. 1D-CNNs use 1D filters for problems involving sequences, which slide along 1D. 1D-CNNs use array multiplication instead of matrix multiplication, which is more computationally efficient and more suited for real-time processing than 3D and 2D-CNNs. Additionally, 1D-CNNs are being proposed for several real-time applications due to their straightforward and compact designs [6], [7].

The number of layers of CNN causes its computational complexity to rise. Our research aims to investigate computationally less expensive methods for classifying IoT and non-IoT devices from their network traffic and assessing the performance of those models. Therefore, since shallow 1D-CNN models require less processing, our work focuses on examining them. They are deemed ideal for real-time processing and more complicated than deep CNNs. We would also assess the model's performance, accuracy, and other categorization metrics as part of our research.

II. RELATED WORK

Here, we include the works that have classified IoT and non-IoT devices using statistical and network flow properties acquired from network traces. The statistical features associated with network traffic flow data, such as packet length, packet rate, and duration of the flow, along with network flow level features like IP addresses, TCP/UDP port numbers as so on [8] are the core focus of statistical and network flow features-based categorization techniques.

Numerous studies have examined and extracted various statistical and network traits from the network traffic flows of IoT and non-IoT devices, and subsequently they have developed features based on those traits for categorization. Statistical factors including data rates, traffic bursts, device activity, and control signalling patterns from traffic flow were used by Sivanathan et al. to classify IoT and non-IoT devices [9]. In a smart environment that was created for the experiment, network traffic data was gathered. The dataset has been made accessible to everyone by Sivanathan et al. 20 IoT devices'

network activity traces made up the dataset. According to the number of servers they communicated with and the application layer protocols they utilized, authors examined the traffic patterns of various IoT devices. A 10 fold cross-validation (10-CV) [9] was adopted to divide the dataset into train and test datasets at a ratio of 9:1. They could obtain 95% accuracy by using a random forest classifier [10]. The precise features/attributes used in the model are not, however, highlighted in the paper.

Similar research by Meidan et al. [11] employed multi-stage meta-classifiers to categorize IoT and non-IoT devices based on network properties collected from TCP traffic. A private dataset of 9 IoT and 4 non-IoT devices was used for this. For their initial round of classification, they used a per device classifier. The flow sessions coming from the IP address are verified to be coming from the device itself using the per device classifier. The second stage classifier, which is used to categorize the IoT device, receives input from the first stage classifiers. The best accuracy archived was 99.281%. Gradient Boosting Machine (GBM) [12], random forest [10], and eXtreme Gradient Boost (XGBoost) [13] was used by the authors. Their work was restricted to TCP flows, though.

Bai et al. in [14] proposed an end-to-end classification method where a segmentation methodology was employed to obtain the statistical network properties. With the use of time stamps to generate the features, the traffic segmentation method was used to divide the traffic flow into smaller traffic flows depending on a predetermined time period. They had used a real-world dataset that Sivanathan et al. had made available to the public [9]. Traffic volume, packet length, network protocols, and traffic direction are among the features that are extracted. By experimenting with a Long Short-Term Memory (LSTM) [15] and CNN architecture, DL methods were investigated. They achieved a classification accuracy rate of 74.8%. IoT devices were also divided into classes in this paper by the authors, including hubs, electrical devices, cameras, switches, and triggers. Non-IoT devices were not taken into consideration because their work was restricted to the identification of IoT devices.

One study that prioritized classification effectiveness and feature/attribute extraction costs was [16]. The dataset provided by Sivanathan et al. was also utilised in this paper too. They were tasked with classifying network traffic and identifying IoT devices. They chose a subset of features based on their predictive power using the CfsSubsetEval algorithm [17]. They selected the four most effective features for the model training: total bytes in the forward and backward directions, maximum packet size in the forward direction, and average packet size in the reverse direction. Less complex features that could be readily collected from the majority of network switches were employed for the model. They were able to classify IoT and non-IoT devices with a 99% accuracy rate by employing the random forest [10] classifier. However, their categorization system limited the IoT devices to 7 classes by classifying them into hubs, cameras, switches, air quality sensors, health, light, and electronics groups.

The researchers in [9] placed a high value on the effectiveness of real-time classification and the expense of feature/attribute extraction. The authors made a portion of the dataset accessible to the public. Their dataset includes network traces that were gathered over the course of six months from 28 distinct IoT and non-IoT devices. They had applied statistical attributes obtained from features of network traffic in their work. They employed a multi-stage machine learning classifier to categorize IoT devices, with the first stage using numerous Naive Bayes (NB) [18] based classifiers and the second stage using a random forest classifier. The cost of extracting the features in real time was divided into three categories by the authors: low cost, medium cost, and high cost. They also demonstrated that they could reach 97.85% accuracy using only low-cost features, with a somewhat higher Root Relative Squared Error (RRSE) of 18.63%. Their analysis, however, did not classify non-IoT devices.

In order to derive the features in real time, Salman et al. employed a different methodology [19]. They included feature extraction, IoT device identification, traffic type identification, and intrusion detection in their framework. The paper's main topic was the IoT security problem. We are concentrating on the IoT device identification step, though. They conducted their research using a private dataset made up of 7 IoT devices. They used 16 packets from a flow to extract information about the flow, including the direction, size, inter-arrival time, and transport protocol, which were then input to the classifier. Even though the flow attributes that are obtained are computationally less expensive, maintaining a list of the packets that make up a session is a burden on network switches' state maintenance. They demonstrated an accuracy of 94.5% for classifying IoT devices into 7 categories using a random forest classifier.

In more recent research, Hameed et al. [20] combined statistical and network aspects. They regarded time-to-live (TTL), MAC addresses, layer-4 protocol numbers, source and destination IP addresses, as network properties. They employed statistical parameters such inter-arrival time, mean packet size, flow rate, and flow burst rate. They used Bag-of-Words (BoW) [21] model as the first step in the transformation of the nominals, including MAC addresses, IP addresses, and port numbers. The research work employed the same subset of the dataset made available by Sivanathan et al. and took into account 21 different IoT devices. Hameed et al. used a multi-stage classifier with a gradient boost technique in the second stage and logistic regression in the first stage to reach an outstanding 99% classification precision. They largely neglected non-IoT devices and concentrated on classifying IoT devices. Their work included source and destination MAC addresses as features that can do one-to-one mapping to IoT devices, and hence can introduce bias. Additionally, these features might not be applicable in Layer 3 environments seen in data centers where Network Address Translation (NAT) is employed.

Shahid et al. used a different method to determine the features for the classification [22]. They employed the t-Distributed Stochastic Neighbour Embedding (t-SNE) tech-

nique to study the network traffic characteristics of the IoT devices, which allowed researchers to pinpoint the attributes that would have the most impact on classification. They had made use of a personal dataset that had been collected from 4 IoT devices linked to a basic home network. Four class labels were determined using the random forest classifier: camera, sensor, lightbulb, and plug. Authors' technique with 10 features demonstrated an outstanding accuracy of 99.9%. Their research overlooked non-IoT devices, and the devices they utilized in their experiment to classify the devices were quite few.

The majority of earlier research was on identifying and categorizing IoT devices based on their semantic properties, meaning, functionality, and usage domain. In [23], a novel strategy was put forth that made use of the Coefficient of Variance (Cu) of the data transmitted to received ratio. IoT devices have been divided into 4 classes. The degree of predictability of device behavior was described by several Cu thresholds, which were then used for categorization [23], [24]. The dataset was made available to the public, and writers set up a smart home scenario in the lab with 41 distinct IoT gadgets. Information Gain (IG) measure was utilized to filter the features for model training. Through experimentation, they settled on 13 different features for their ultimate model. With 41 different IoT devices, they employed an ensemble LogitBoost model, which produced a very high classification accuracy of 99.79%.

1D-CNN has recently been utilized for categorization. 1D-CNNs have excelled in a variety of applications, as demonstrated by [7]. The applications where 1D-CNNs have been tested and demonstrated success are described below. Examples include automatic speech recognition (ASR) [25], real-time electrocardiogram (ECG) monitoring [26], and vibration-based structural damage detection in civil infrastructure [27]–[31].

In several applications, 1D-CNNs have attained state-of-the-art results in terms of accuracy and performance (computational cost). Less sophisticated models that can be utilized for real-time classification and perform effectively are required for the deployment of frameworks for the classification of IoT and non-IoT devices. 1D-CNN models have recently grown in popularity due to their simplicity, compactness, and lower computational cost while yet being able to achieve excellent classification accuracy. To the best of our knowledge, we are unaware of any research projects that have classified IoT and non-IoT devices using 1D-CNN. The proposed classification task is highly suited for 1D-CNN models since they can detect non-axis parallel decision boundaries and capture sequences (for instance, a flow of packets). Additionally, it may create classifiers that are incredibly accurate. For our research, we employed shallow 1D-CNN because it has a lower computational cost than deep CNN. Our research also takes into account device-to-device data transfers through TCP and UDP.

III. RESEARCH METHODOLOGY

Our study's main objective is to identify IoT and non-IoT devices in network traffic in real time. This might be installed at edge data center switches, allowing network operators to manage the available network resources effectively and efficiently while accommodating the various traffic demands made by these devices.

A. Dataset

The University of New South Wales (UNSW) Sydney's publicly accessible dataset [9], was used in our research. The data set was gathered from a university-set up smart environment lab. The data set comprises of raw Packet Capture (PCAP) network traffic data that was acquired over a 20-day period from 29 distinct IoT and non-IoT devices. All of the devices that were used in the configuration are listed in Table I.

TABLE I
IoT AND NON-IoT DEVICES

IoT devices	Non-IoT devices
Smart Things	Samsung Galaxy Tab
PIX-STAR Photo-frame	Android Phone
iHome	IPhone
Withings Smart scale	MacBook/IPhone
HP Printer	Laptop
Amazon Echo	MacBook
Dropcam	Laptop
Blipcare Blood Pressure meter	
TP-Link Day Night Cloud camera	
Insteon Camera	
Dropcam	
Nest Dropcam	
Withings Smart Baby Monitor	
Samsung SmartCam	
Netatmo Welcome	
Belkin wemo motion sensor	
Light Bulbs LiFX Smart Bulb	
TP-Link Smart plug	
Netatmo weather station	
HP Printer	
NEST Protect smoke alarm	
Withings Aura smart sleep sensor	

The dataset captures data traffic from user and application requests as well as control communication between devices. Over the course of 20 days, the dataset was gathered and organized on a daily basis. The device identifiers used as class labels in the presented data set are Media Access Control (MAC) addresses, which are specific to each device.

B. Preprocessing

The dataset is made up of 20 raw PCAP files that were used to capture the network traces of all 28 devices over the course of 20 days. The raw PCAP files must be divided into PCAP files for each session that record the sequence of network packet exchanges between the device pairs during the course of the conversation. This phase assists in identifying the conversation because the network packet traces reveal the device pair involved in the communication. A session is a period of time during which a device pair can communicate in

both directions. A combination of five tuples, including source IP, destination IP, transport protocol, (UDP/TCP) source, and destination port, uniquely identifies each session. The SplitCap utility [32] is used to divide PCAP files into individual sessions. The SplitCap utility divides a PCAP file into several PCAP files that each capture a sequence of network packet exchanges across device pairs during the course of a communication. In the network packet of the session, the MAC addresses of the IoT and non-IoT devices are used to label classes.

For further research and model building, essential statistical and network traffic flow features are retrieved from the per-session PCAP files processed by the SplitCap tool. The Canadian Institute of Cyber Security at the University of New Brunswick created CICFlowMeter [33], a piece of software that is used to extract flow level features from the raw PCAP files. The application extracts 84 features at the flow level from raw PCAP files and saves them as Comma Separated Values (CSV) files.

C. Cleaning

The CSV dataset derived from the CICFlowMeter utility includes data on control flows like

- Broadcast and multicast related communication.
- Network traffic exchanged by router/gateway.
- Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP) and other control network traffic.

Control flow details are not significant to this investigation because they do not provide any information about categorization jobs. These data are taken out of the dataset. The CSV dataset created by CICFlowMeter also contains the following features, but they were omitted because the study does not apply to them.

- **Flow ID:** It is an identifier generated by CICFlowmeter.
- **Timestamp:** It is derived from the first packet of the session.
- **Source IP:** When a device is prepared for communication, this feature determines the IP allocated to the IoT or non-IoT device. This functionality has been removed since it could lead to bias mapping specific to the device. Additionally, when using Network Address Translation (NAT) techniques in the network and in Layer-3 environments at edge data centers, this feature may not be available.
- **Source Port:** The port number assigned by the originating program or device is identified by this feature. Again, this feature can be minor due to methods like NAT, as they are translated by routers and gateway switches.
- **Feature set:** Forward (Fwd) Push (PSH) Flags, Backward (Bwd) Push (PSH) Flags, Fwd Urgent (URG) Flags, Bwd Urgent (URG) Flags, Finish (FIN) Flag Count, Synchronize (SYN) Flag Count, Reset (RST) Flag Count, Push (PSH) Flag Count, Acknowledge (ACK) Flag Count, Urgent (URG) Flag Count, Congestion Window-Echo (CWE) Flag Count, Early

Congestion-Echo ECE Flag Count, Subflow Fwd Pkts, Subflow Bwd Pkts, Subflow Fwd Byts, and Subflow Bwd Byts: These characteristics are exclusive to TCP. These attributes aren't included in the dataset because our research focuses on managing network traces for the UDP and TCP protocols.

- **Feature set: Fwd Byts/block Average, Fwd Pkts/block Average, Fwd Block Rate Avg, Bwd Byts/block Average, Active Mean, Active Standard Deviation, Active Maximum, Active Minimum:** These features have only zero values, hence they cannot encode any valuable information.

D. Feature Sets

For our study, we created four different sets of features to use in the construction of 1D-CNN classifiers and the performance testing of those classifiers. Prior to that, the dataset is cleaned by removing information about pointless features. After cleaning, every feature is present in the initial feature set. The complexity of extracting/deriving the features from network routers/switches in terms of computing and memory overhead determines the construction of the other three sets. For convenience, the four separate feature sets are referred to as FS-1, FS-2, FS-3, and FS-4. The arrangement of feature sets is based on diminishing levels of relative complexity. The least complex of the four is FS-4, whereas FS-1 is the most complex. There are 55, 29, 21, 15, and 15 features in FS-1, FS-2, FS-3, and FS-4, correspondingly. The various features picked for each FS collection are displayed in Table II.

E. 1D-CNN Model

Our main objective is to investigate an appropriate shallow 1D-CNN model for real-time classification. Four sets of flow spatio-temporal features collected from the devices' flow traces were used to train and assess 1D-CNN models. The model is adjusted by changing several hyper-parameters for the best outcomes (such as filter sizes, number of convolution layers, etc.).

The architecture of a shallow 1D-CNN that is investigated in our research is shown in Fig. 2. The model consists of two convolutional layers and two pooling layers, followed by a dense layer and an output layer, as shown in the figure.

In our study, four distinct 1D-CNN models were created and each was fed with the FS-1, FS-2, FS-3, and FS-4 feature sets. For feature sets FS-1 through FS-4, the first convolutional layer has 55, 29, 21, and 15 neurons, respectively. There are 2592, 2592, 2592, and 64 neurons in the second convolutional layer, first max pooling layer, second max pooling layer, and dense layer, respectively. Rectified Linear Unit (ReLU) [34] activation function was employed in our research for convolution and dense layers. Before the last softmax layer, we employed a single dense layer. The dense layer would use ReLu activation function and be maintained with minimum input neurons. 64 neurons were discovered to be the ideal number of neurons in a dense layer during studies. A SoftMax

TABLE II
FEATURESETS

FS-1	FS-2	FS-3	FS-4
Flow Duration, Idle Mean, Fwd Act Data Pkts, Init Bwd Win Byts, Flow Byts/s, Idle Min, Bwd Pkt Len Mean, Dst Port, Flow IAT Max, Fwd Pkt Len Min, Init Fwd Win Byts, Pkt Len Max, Fwd Seg Size Min, Fwd Pkt Len Mean, Pkt Len Std, Bwd IAT Max, Down/Up Ratio, Fwd Pkts/s, Fwd Pkt Len Max, Idle Max, Flow Pkts/s, Bwd Seg Size Avg, Idle Std, Bwd Pkt Len Std, Fwd Seg Size Avg, Protocol, Fwd IAT Mean, Tot Fwd Pkts, TotLen Bwd Pkts, Flow IAT Std, Tot Bwd Pkts, Pkt Len Min, Pkt Len Mean, Bwd IAT Tot, Bwd IAT Mean, Fwd Pkt Len Std, Bwd Pkt Len Min, sBwd Pkts/b Avg, Fwd IAT Max, Pkt Len Var, TotLen Fwd Pkts, Dst IP, Bwd Header Len, Bwd Blk Rate Avg, Pkt Size Avg, Flow IAT Min, Flow IAT Mean, Fwd IAT Min, Bwd IAT Std, Bwd Pkt Len Max, Bwd IAT Min, Fwd IAT Tot, Fwd IAT Std, Bwd Pkts/s, Fwd Header Len	Fwd Pkt Len Min, Protocol, Pkt Size Avg, Bwd Pkt Len Mean, Fwd Pkt Len Mean, Fwd Pkts/s, Flow Byts/s, Fwd Header Len, Pkt Len Mean, TotLen Bwd Pkts, Dst Port, Fwd Pkt Len Max, Fwd Pkt Len Std, Pkt Len Var, Flow Pkts/s, Tot Fwd Pkts, Bwd Pkts, Bwd Pkts/s, TotLen Fwd Pkts, Tot Bwd Pkts, Pkt Len Max, Dst IP, Fwd Act Data Pkts, Pkt Len Min, Bwd Pkt Len Min, Bwd Pkt Len Max, Bwd Header Len, Flow Duration, Bwd Pkt Len Std, Pkt Len Std	Protocol, Flow Byts/s, Pkt Size Avg, Flow Pkts/s, Pkt Len Min, Bwd Header Len, Pkt Len Var, Dst IP, Bwd Pkts/s, Pkt Len Max, Tot Fwd Pkts, Dst Port, Fwd Act Data Pkts, Tot Bwd Pkts, TotLen Fwd Pkts, Fwd Pkts/s, Pkt Len Mean, Fwd Header Len, TotLen Bwd Pkts, Flow Duration, Pkt Len Std	Bwd Pkts/s, TotLen Bwd Pkts, Fwd Header Len, Dst IP, Tot Fwd Pkts, Dst Port, Fwd Pkts/s, Protocol, Flow Duration, Tot Bwd Pkts, Fwd Act Data Pkts, Flow Byts/s, TotLen Fwd Pkts, Flow Pkts/s, Bwd Header Len

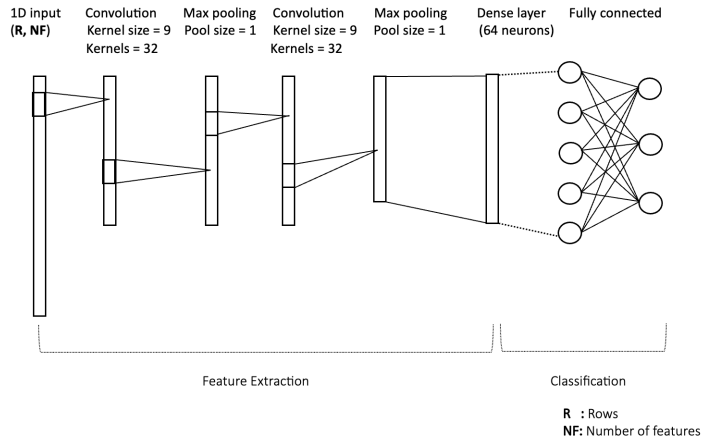


Fig. 2. The proposed shallow 1D-CNN

function is utilized at the output layer since our classification task involves multi-class classification and produces a probability distribution on the device classes. As our objective function, we also employed the categorical cross entropy loss [35]. There are one class for non-IoT devices and 22 classes used to categorize various IoT devices. As a result, the softmax layer uses a total of 23 classes as output. Epochs ranging from five to twenty were used in all tests, with ten being the most effective. In our tests, stochastic optimization was carried out using the Adam optimizer [36]. Adam is an adaptive learning rate optimizer created exclusively for neural network training. Adam optimizers are renowned for their minimal memory footprint and low computational cost, both of which will be beneficial for training the dataset. Ten epochs were discovered to be the most effective out of all the experiments, which used epochs ranging from five to twenty.

IV. MODEL EVALUATION

Different classification metrics that are available in the literature to comprehend the success of the categorization and its trade-offs with regard to real-time deployments, the 1D-CNN model is assessed. Some of these are Classification Accuracy, Precision, and True Positive Rate (TPR or Recall) [37], KAPPA [38], and F-Measure [39]. Among these we used Classification Accuracy here.

A. Hyperparameters tuning

Following are the hyper parameters were experimented and tuned for optimal results.

- Number of convolution/pooling layers
- Batch size
- Kernel/filter size
- Number of filters
- Pooling size
- Number of kernels/filters

Table III lists the optimal hyperparameters results

V. MODEL RESULTS

The classification report for each feature set produced by our model is provided in Table IV. K-Fold cross validation with K=10 was applied to the dataset during training. For each kth fold step, ten continuous iterations (epochs) of the validations were performed.

It is clear from Table IV that FS-1 with 55 characteristics performed best, scoring almost 97% test accuracy. Another intriguing point to consider is that FS-4, which has only 15 features, has likewise performed admirably well, reaching 94% test accuracy while losing only 3% ground to FS-1. Looking more closely at the characteristics present in FS-4, the following general categories can be made:

- **Network layer features - (Dst IP, Dst Port, Protocol):** All routers and switches extract these features in order to forward traffic. Since they may be easily employed for the classification process, there is no additional cost associated with collecting these features from packets.

TABLE III
OPTIMAL HYPERPARAMETERS

Hyperparameter	Optimal value	Remarks
Convolution and pooling layers	Two layers	Tests have revealed that the performance of categorization did not significantly improve with the addition of the third layer. Therefore, two layered arrangement was determined to be the optimal hyper parameter for reducing complexity
Kernel/filter size	Nine	Nine was a good kernel size for classification accuracy, according to experiments. After nine, there was little to no noticeable change. Nine kernels were therefore determined to be the ideal hyper parameter
Number of kernels/filters	32	Studies have showed that there was little gain in classification accuracy with more kernels or filters than 32. A trade-off exists between increasing kernel size and classification accuracy. 32 kernels were chosen as the optimal hyper parameter to balance the tradeoff without significantly reducing the model complexity
Pooling size	One	To maintain the convolution output without reducing the dimension, the pooling size was set to one. Higher pool sizes led to decreased accuracy. Given that the dataset employed has smaller feature dimensions, dimensionality reduction did not play a significant role in the model's complexity reduction. Pool size of 1 was selected as the optimal hyper parameter to achieve the desired balance between model complexity and accuracy
Batch size	32	Batch size 32 had demonstrated a reasonable balance between improved accuracy results and training duration out of all the batch sizes tested. Consequently, 32 was chosen as the optimal hyper parameter for batch size

- **Packet level statistics features (Fwd Act Data Pkts, Tot Fwd Pkts, TotLen Fwd Pkts, Tot Bwd Pkts, Fwd Header Len, TotLen Bwd Pkts, Bwd Header Len, Fwd Pkts/s, Bwd Pkts/s,):** The majority of data center routers and switches keep the aforementioned statistics for accounting purposes. Consequently, these can be used for classification with ease.
- **Flow level statistics features - (Flow Byts/s, Flow Pkts/s and Flow Duration):** These properties can be easily inferred from flow table approaches for routers and switches that use them to forward network traffic. This requires flow state to be preserved for the extraction of these features.

TABLE IV
CLASSIFICATION REPORT

Feature Set	Features	Metric	Score
FS-1	55 features	Training accuracy	96.7%
		Test accuracy	96.6%
		KAPPA score	96.2%
		Weighted average precision	96.7%
		Weighted average recall	96.7%
		Weighted average F1-score	96.7%
FS-2	29 features	Classification time for single test record	34 μ secs
		Training accuracy	96.5%
		Test accuracy	96.2%
		KAPPA score	96.3%
		Weighted average precision	96.3%
		Weighted average recall	96.3%
FS-3	21 features	Weighted average F1-score	96.3%
		Classification time for single test record	31 μ secs
		Training accuracy	96.4%
		Test accuracy	96.2%
		KAPPA score	95.7%
		Weighted average precision	96.3%
FS-4	15 features	Weighted average recall	96.2%
		Weighted average F1-score	96.2%
		Classification time for single test record	31 μ secs
		Training accuracy	94%
		Test accuracy	93.8%
		KAPPA score	92.9%
		Weighted average precision	94%
		Weighted average recall	93.8%
		Weighted average F1-score	93.8%
		Classification time for single test record	31 μ secs

In light of this, it can be said that the 1D-CNN model can classify IoT and non-IoT devices with as high as 94% classification accuracy utilizing as few as 4 network flow variables, 12 packet and byte-level statistical data, and 3 network level features. In conclusion, it can be shown that the 1D-CNN model only required 34 μ secs and 31 μ secs, respectively, for a single classification on FS-1 and FS-4, as shown in Table IV.

A. Results comparison

Here, we compare the outcomes of our work to earlier papers in the field of relevant research. For our research, Sivanathan et al. made 20 days' worth of data available to the public. The comparison focuses on previous related articles that have used comparable datasets. The key similarities are highlighted in Table V.

In order to create the dataset Sivanathan et al., [9] had to collect data over the course of six months. However, just a portion of the traffic data (20 days) was made available to the

TABLE V
RESULTS COMPARISON

Author(s)	Accuracy	Remarks
Sivanathan et al., 2019 [9]	97.85%	Only a portion of the dataset 20 days was made available to the public after the authors used the entire six months of the dataset for their research.
Hameed & Leivadeas, 2020 [20]	99%	MAC addresses were one of the features they used for classification. The one-to-one mapping of IoT devices that MAC addresses are capable of creating bias when used as features. At edge routers and switches, where traffic is aggregated, MAC addresses of IoT and non-IoT devices could not be accessible.

general public. With this small dataset, we have conducted our work. A precise comparison is therefore impossible. Although the dataset is small, our suggested model is nevertheless able to provide a 97% accuracy rate, which is pretty equivalent to the 97.85% accuracy rate attained by Sivanathan et al.

Using a comparable dataset, Hameed & Leivadeas [20] who were able to reach 99% classification accuracy. However, the source and destination MAC addresses were employed by the authors as features in their model. As these attributes can map one-to-one to IoT devices, this may introduce bias. In addition, these functions may not be available in Layer-3 contexts and with address translation methods like NAT in network summarizers such data center router/switches. We did not use MAC addresses or source IP addresses as classification criteria in our analysis.

VI. CONCLUSION

In this research, we used statistical and network flow data from the network traces to create a shallow 1D-CNN model. The classification of IoT and non-IoT devices using TCP and UDP traffic flows was done using the model. The shallow 1D-CNN model worked effectively, reaching a high classification accuracy of 97%, according to the results. The outcomes also demonstrate that 1D-CNNs are capable of achieving good classification accuracy with just two layers of convolution and pooling layers.

Four separate feature sets, numbered from FS-1 through FS-4, were tested. Based on the relative difficulty of extracting/deriving the features from network routers/switches, these feature sets were selected. The FS-1 feature set, which comprises 55 features, is the most complicated of the four feature sets. The least difficult feature set is FS-4, which has 15 features. With just 15 statistical and network flow variables, we were able to demonstrate that a shallow 1D-CNN is capable of achieving as high as 94% classification accuracy. The model may obtain a classification accuracy of up to 97% using 55 different statistical and network flow variables.

REFERENCES

- [1] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in IoT networks: A survey," *Journal of Network and Computer Applications*, vol. 154, p. 102538, 2020.

- [2] H. Chang, A. Hari, S. Mukherjee, and T. Lakshman, "Bringing the cloud to the edge," in *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 346–351, IEEE, 2014.
- [3] P. P. Ray, "A survey on internet of things architectures," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 3, pp. 291–319, 2018.
- [4] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.
- [5] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *Journal of Systems Engineering and Electronics*, vol. 28, no. 1, pp. 162–169, 2017.
- [6] F. U. Islam, G. Liu, J. Zhai, and W. Liu, "VoIP traffic detection in tunneled and anonymous networks using deep learning," *IEEE Access*, vol. 9, pp. 59783–59799, 2021.
- [7] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *Mechanical systems and signal processing*, vol. 151, p. 107398, 2021.
- [8] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM transactions on networking*, vol. 23, no. 4, pp. 1257–1270, 2014.
- [9] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying IoT devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2018.
- [10] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [11] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "ProfilIoT: A machine learning approach for IoT device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, pp. 506–509, 2017.
- [12] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.
- [13] T. Chen, "Story and lessons behind the evolution of xgboost," 2016.
- [14] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang, "Automatic device classification from network traffic streams of internet of things," in *2018 IEEE 43rd conference on local computer networks (LCN)*, pp. 1–9, IEEE, 2018.
- [15] A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.
- [16] M. R. Santos, R. M. Andrade, D. G. Gomes, and A. C. Callado, "An efficient approach for device identification and traffic classification in IoT ecosystems," in *2018 IEEE Symposium on Computers and Communications (ISCC)*, pp. 00304–00309, IEEE, 2018.
- [17] M. A. Hall, "Correlation-based feature subset selection for machine learning," *Thesis submitted in partial fulfillment of the requirements of the degree of Doctor of Philosophy at the University of Waikato*, 1998.
- [18] A. McCakum, "Graphical models, Lecture2: Bayesian network representation," 2019.
- [19] O. Salman, I. H. Elhajj, A. Chehab, and A. Kayssi, "A machine learning based framework for iot device identification and abnormal traffic detection," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3743, 2022.
- [20] A. Hameed and A. Leivadadas, "IoT traffic multi-classification using network and statistical features in a smart environment," in *2020 IEEE 25th international workshop on computer aided modeling and design of communication links and networks (CAMAD)*, pp. 1–7, IEEE, 2020.
- [21] Q. Wang, J. Xu, H. Chen, and B. He, "Two improved continuous bag-of-word models," in *2017 international joint conference on neural networks (IJCNN)*, pp. 2851–2856, IEEE, 2017.
- [22] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "IoT devices recognition through network traffic analysis," in *2018 IEEE international conference on big data (big data)*, pp. 5187–5192, IEEE, 2018.
- [23] I. Cvitic, D. Perakovic, M. Perisa, and M. Botica, "Definition of the IoT device classes based on network traffic flow features," in *4th EAI International Conference on Management of Manufacturing Systems*, pp. 1–17, Springer, 2020.
- [24] I. Cvitic, D. Perakovic, M. Perisa, and M. D. Stojanovic, "Novel classification of IoT devices based on traffic flow features," *Journal of Organizational and End User Computing (JOEUC)*, vol. 33, no. 6, pp. 1–20, 2021.
- [25] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 10, no. 22, pp. 1533–1545, 2014.
- [26] S. Kiranyaz, T. Ince, R. Hamila, and M. Gabbouj, "Convolutional neural networks for patient-specific ECG classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2608–2611, IEEE, 2015.
- [27] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-D convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7067–7075, 2016.
- [28] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks," *Journal of Sound and Vibration*, vol. 388, pp. 154–170, 2017.
- [29] O. Abdeljaber, O. Avci, M. S. Kiranyaz, B. Boashash, H. Sodano, and D. J. Inman, "1-D CNNs for structural damage detection: Verification on a structural health monitoring benchmark data," *Neurocomputing*, vol. 275, pp. 1308–1317, 2018.
- [30] O. Avci, O. Abdeljaber, S. Kiranyaz, B. Boashash, H. Sodano, and D. J. Inman, "Efficiency validation of one dimensional convolutional neural networks for structural damage detection using a SHM benchmark data," in *Proceedings of 25th International Congress of Sound and Vibration (ICSV)*, pp. 4600–4607, 2018.
- [31] S. Kiranyaz, A. Gastli, L. Ben-Brahim, N. Al-Emadi, and M. Gabbouj, "Real-time fault detection and identification for MMC using 1-D convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8760–8771, 2018.
- [32] J. Kotak and Y. Elovici, "IoT device identification using deep learning," in *13th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2020) 12*, pp. 76–86, Springer, 2021.
- [33] I. Cvitic, D. Perakovic, M. Perisa, and B. Gupta, "Ensemble machine learning approach for classification of IoT devices in smart home," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 11, pp. 3179–3202, 2021.
- [34] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," *arXiv preprint arXiv:1803.08375*, 2018.
- [35] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.
- [36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [37] T. Eftestøl, "Controlling true positive rate in ROC analysis," in *2009 36th Annual Computers in Cardiology Conference (CinC)*, pp. 353–356, IEEE, 2009.
- [38] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.
- [39] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, F-score and ROC: A family of discriminant measures for performance evaluation," in *Australasian joint conference on artificial intelligence*, pp. 1015–1021, Springer, 2006.