International Journal of Information Security (2026) 25:5
https://doi.org/10.1007/s10207-025-01171-4

REGULAR CONTRIBUTION

IoT VisPerNet: Adversarial Perturbation Visualization for IoT Networks

Jonathan Gregory[1] · **Jacob Auerbach**[1] · **Qi Liao**[1]

Received: 27 January 2025 / Accepted: 23 November 2025

Abstract

Network Intrusion Detection Systems (NIDS) are vital for the security of modern networks, especially for networks that contain vulnerable IoT devices. While NIDS can use robust artificial intelligence (AI) / machine learning (ML)-based network traffic classifiers to better recognize and defend against traffic that represents network attacks and intrusions, these models exhibit black-box behaviors and thus the security of these AI/ML-based security mechanism is largely unknown. This research investigates the impacts of adversarial IoT NetFlow traffic crafted using perturbations generated by a custom variant of the Fast Gradient Signed Method on a CNN-based network traffic classifier. This research also creates a novel, custom tool for visualizing these adversarial traffic examples using statistical tools and various visualizers. The resulting tool, IoT VisPerNet, aids in explaining how these adversarial perturbations on IoT NetFlow traffic are effective. Our experiments generate templates of adversarial IoT network traffic that can be studied to learn the weaknesses of machine learning classifiers. The results from this research's perturbation algorithm show promise in fooling the traffic classifier but require improvement to be more fine-grained. Further, IoT VisPerNet is effective in highlighting certain features in the IoT network traffic that greatly affect traffic classification for different attack classes.

Keywords Cybersecurity · Adversarial attack · Visualization · Perturbations · Network Intrusion Detection · Internet of Things (IoT) · Network Traffic classification · Explainable Artificial Intelligence

1 Introduction

Analyzing network traffic and accurately differentiating benign traffic from network attacks – such as distributed / denial of service (DoS/DDoS), network scanning, malware / ransomware, and man-in-the-middle (MITM) attacks – is an essential role of modern network intrusion detection systems (NIDS). The role of a strong, robust NIDS is made all the more important for networks containing Internet of Things (IoT) devices, as many of these devices lack security in their design and deployment and thus represent vulnerable devices – and possible threat vectors – on a network [19]. Given that the number of connected IoT devices is more than double the world's population, it is of pressing importance to research network security for this vast and varied threat surface [18].

Several traditional solutions exist for this problem of network traffic classification in a NIDS. As an existing study [16] notes, signature-based methods for detection – whereby a traffic signature is matched against known benign and attack patterns by the NIDS – are effective at recognizing known attacks, but cannot extrapolate to defend against novel attack patterns. Other methods for traffic classification include packet inspection and port-based classification, but as stated by [8], these methods are ineffective against traffic that uses encryption and dynamic port allocation, respectively. Advances in artificial intelligence (AI) and its subset, machine learning (ML), have provided better, more robust network traffic classification methods that can be employed by a NIDS [18]. For instance, both support vector machines and deep neural networks can be used to combat spoofing attacks on IoT devices, and multilayer perceptrons can be used to prevent DoS attacks [18]. ML models trained on packet flow data for network traffic classification can also effectively classify encrypted traffic by accessing unencrypted flow information such as the source and destination port, number of packets, and duration of the packet flow [8]. As such, this research focuses on ML-based network traffic classification using network flow data as a promising solution to the problem of network intrusion defense.

✉ Qi Liao
liao1q@cmich.edu

[1] Department of Computer Science, Central Michigan University, Mount Pleasant, MI 48859, USA

Published online: 10 December 2025

Although ML-based network traffic classification models are effective against expected attacks, it is important to verify that these commonly named "black box" models behave as expected when faced with adversarial examples, or traffic engineered to deceive the models despite containing defining features characteristic of benign or attack traffic. Given the importance of network intrusion detection, it is vital that perturbations of – or changes to – traffic features that preserve the fundamental purpose of the traffic do not result in traffic misclassification. Further, these models must also be validated and thus made explainable – that is, given varied types of network traffic input, the behaviors of the model must be interpretable in a statistical or logical sense. These requirements for model validation lead to the following research question: "Can AI/ML-based NIDS be fooled using adversarial perturbations on malicious IoT network traffic, and if so, how can the impacts of these perturbations on the traffic be explained to humans?"

To further specialize this research, the source of network traffic used in these experiments is traffic originating from testbed IoT networks. This specialization is necessary to more completely cover the problem of NIDS protection for IoT devices, as although traditional network-connected devices and IoT devices share traffic characteristics that can be monitored by a NIDS, traffic generated by IoT devices differs significantly from traditional traffic. For example, Mainuddin et al. [10] indicate that traffic from specific IoT devices can be confidently differentiated from non-IoT traffic – and even from traffic from other IoT devices – using IP-, TCP-, and flow-related features. Moreover, certain attacks – such as Mirai botnet infections – are specifically related to IoT devices [2]. It is therefore valuable to focus investigations on this IoT traffic and the ways in which perturbations of this traffic affect the confidence of a machine learning network traffic classifier.

This research focuses on developing adversarial traffic samples to deceive a convolutional neural network (CNN)-based IoT network traffic filter using novel adversarial perturbations generated by a variant of the Fast Gradient Signed Method (FGSM) algorithm. Once perturbed, the adversarial network traffic and unperturbed traffic samples are visualized through a custom interactive visualization tool (IoT VisPerNet) to better explain the performance of the model and its reaction to the perturbed network traffic samples. The goal of this research is to better understand how an adversary could feasibly break or trick a robust AI/ML network traffic classifier by manipulating network traffic data, such as packet size and flow duration. As such, this research differs from more traditional studies in that it places no restrictions on how large or small these manipulations of the flow data can be. The underlying goal of this research is to use adversarial machine learning to generate templates of network traffic that are classified as benign by a target defense model despite containing defining characteristics of attack traffic. Only attacker-modifiable fields – such as source port numbers – are affected by the perturbations in this research, thus creating more realistic templates of adversarial traffic that could be sent by adversaries.

The adversarial attack traffic developed in this research succeeded in decreasing the recall of a high-performing CNN-based traffic classifier by 71.7%. Moreover, using the IoT VisPerNet tool to analyze this adversarial network traffic yields interesting observations that may be used to improve future network classifiers. Importantly, IoT VisPerNet helps to make the performance of the model on these adversarial traffic examples more understandable, thus achieving a key goal of explainable artificial intelligence (XAI) / machine learning. The contributions of this research to the fields of network traffic classification and visualization are therefore the application of a custom variant of the FGSM perturbation attack to IoT network traffic and the development of a new interactive visualization tool that can be used to observe the results (model predictions) of these adversarial perturbations on the network traffic in real time. This research is therefore useful in exploring the weaknesses of network traffic classification models, and the insights gleaned from this research can be used to fix these weaknesses in the future.

The remainder of this paper contains the following. First, Section 2 will discuss related work to this problem of network traffic classification, adversarial perturbations on network traffic, and network traffic visualization. Section 3 will discuss the key algorithms used in this research to train the classification model, perturb the network traffic, and build the custom visualization tool, IoT VisPerNet. Section 4 will present and examine the experimental results. Finally, Section 5 will conclude with a discussion of the limitations of this research and future work.

2 Related Work

Varied approaches exist for deep machine learning-based network traffic classification using both packet-level information and flow data. Several notable and effective examples include the Tree-RNN technique developed by Ren et al. [14], which classifies encrypted network traffic using a recurrent neural network (RNN)-based model trained on raw pcap packet captures converted to grayscale images; the C-LSTM model (a hybrid CNN and RNN model) proposed by Kim and Cho [7] for detecting traffic anomalies from time-series network traffic data; the dense neural network (DNN) trained by Long and Jinsong [8] on NetFlow data, a format of network flow data; TFAD model [20] for Software-defined networks (SDN); and the CNN and RNN-based model developed by Lopez-Martin et al. [9] for IoT network traffic classification using network flow data.

CNN-based classifiers are of principal interest in this research due to their robust detection abilities and ubiquity in research. Variations of 1-Dimensional (vector-based) and 2-Dimensional (image- or matrix-based) CNN network traffic classifiers exist that use different inputs and architectures. For example, an early approach for CNN-based network traffic classification was proposed by Wang et al. [22], who used an image-based 2D-CNN model to perform classification on raw packet data converted to grayscale images. Qazi et al. [13] developed an effective 1D-CNN for network intrusion detection using only the given feature vector as input. Since network traffic flow statistics are compact, amenable to machine learning, and, as shown by Santos et al. [15], can be used in an integrated NIDS pipeline, this research focuses on these flow statistics, particularly in NetFlow format.

Previous research has investigated crafting adversarial perturbations of network traffic using a variety of methods. In particular, many of these studies use Goodfellow et al.'s [4] Fast Gradient Signed Method (FGSM), a white-box attack that uses the sign of the gradients of the cost function of a classifier with respect to an input x to craft an adversarial example x'. FGSM-based attacks are fast, cheap, and effective against neural networks that behave linearly and especially CNN-based classifiers, making them of interest in this research. Hu et al. [6] used FGSM and other perturbation methods to create perturbations on traffic data converted to images to attack a classification pipeline like the one described in classification work [22], but Hu et al. did not account for whether these perturbations corrupt the integrity and functionality of the original traffic. Alhajjar et al. [1] also investigated perturbations on network traffic data, and while the authors did not use FGSM, the perturbations do preserve the functionality of the traffic being perturbed by making certain fields in network traffic flow data – such as TCP flags – immutable and by applying the condition that changes to network features can only be additive. A hybrid approach of these two methods – using FGSM to perturb network traffic and making certain fields of this traffic immutable to perturbations – is adopted in this research. Unlike in work [1], perturbations to the mutable fields are not restricted to only be positive, as the goal of this research is to intelligently generate templates of what adversarial traffic that deceives a traffic classifier *could* look like according to our machine learning methods. A similar perturbation approach [5] was applied with success to word vectors in spam, and the constrained nature of the domain in this research makes the earlier algorithm promising for generating finer-grained, adaptively scaled perturbations in this research.

Understanding how a machine learning model makes its decision has become increasingly relevant as organizations try to address business, ethical, and legal concerns when using AI and ML, a concept closely related to explainable artificial intelligence (XAI) [11]. This is also a prevalent concern when trying to understand what type of malicious traffic an AI/ML-based NIDS is susceptible to. This problem has been addressed to some extent in past research using visualization. For example, a previous study [17] tried to understand unsupervised clustering algorithms using a scatter plot visualization technique. While research in understanding a ML model is relatively new, much more extensive research has been done on data visualizations and how to pick the best visualization to use for specific tasks. Visualizations do not have a one-size-fits-all view for all types of data. Therefore, multiple different types of views are often needed to understand given data. Several studies including [23] and [3] have looked at different views, specifically focusing on multidimensional data. Their visualizations include star diagrams, scatter-plot matrices, parallel coordinates, tree maps, and pie charts.

This research proposes different visualizations that can be used with multiple features to enhance their use. To improve scalability, the method of brushing is introduced to allow users to select the sections of perturbation data they want to see and not just view thousands of data points at once. To get a better idea of what type of data the ML model is misclassifying, the visualization tool in this research also uses a box and whisker plot similar to the one in [21]. This view enables the user to see insightful trends of values the model tends to miss on average for each feature. IoT VisPerNet enhances interpretability and explainability by enabling users to visualize patterns across numerous traffic flows, rather than focusing solely on isolated feature attributions as in traditional XAI tools like LIME or SHAP. VisPerNet allows users to observe which features remain stable or vary across attacks, how frequently they change, and by how much. Unlike LIME/SHAP, which highlight feature importance without detailing value-level impacts, VisPerNet provides actionable information that can be used to generate new traffic samples emphasizing anomalous or underrepresented values to improve model robustness.

3 VisPerNet Architecture

In this section, we outline the key methodologies and algorithms used to conduct adversarial machine learning attacks and a visualization tool designed to analyze perturbation insights in generated adversarial IoT network traffic samples. Figure 1 presents the architecture overview of IoT VisPerNet system. A system operator or administrator deploys a Network Intrusion Detection System (NIDS) equipped with a machine learning-based traffic classifier to monitor and filter IoT network traffic. An attacker (or the operator when aiming to assess and enhance the NIDS's robustness) may employ adversarial machine learning techniques, introducing various perturbations to generate malicious traffic that the

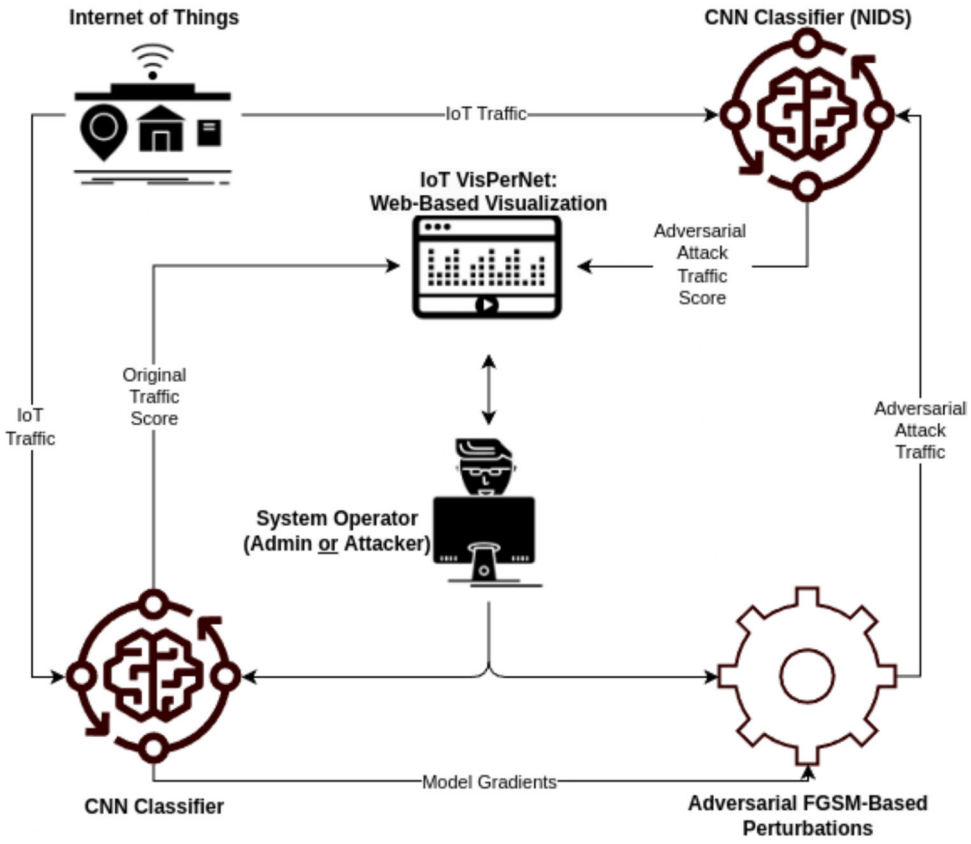

Fig. 1 Architecture overview of the IoT VisPerNet network traffic adversarial perturbation and visualization

NIDS mistakenly identifies as benign. This process, including the comparison of original and perturbed traffic samples alongside prediction scores, is thoroughly analyzed using an interactive web-based visualization tool.

3.1 Adversarial Perturbation Algorithms

Features of typical network traffic in the format of NetFlow may include protocols, source and destination IP address, port numbers, TCP flags, bytes, TTLs, etc. It is important to note that modifications to certain fields in the traffic data can possibly change the nature of traffic. To enhance the realism and difficulty of adversarial attacks, we assume that not all features of network traffic are modifiable. For instance, while an attacker can freely select source port numbers, they cannot alter destination port numbers or protocols, as doing so would prevent the traffic from reaching the intended target hosts. It is also understood that outgoing traffic captured by a NIDS must originate from a server protected by the NIDS, so the attacker cannot directly modify attributes of outgoing traffic. A comprehensive list of immutable features in a typical IoT network is provided in Table 1, thus these features were not affected by adversarial perturbations.

The main methodology to generate adversarial IoT network traffic in this study is Fast Gradient Signed Method

Algorithm 1 Adaptive Adversarial Perturbations on IoT Network Traffic

Input: M, the ML classifier; X, the sample input; L, the input's target label; V, an array of minimum and maximum feature values; P, the set of the indices of immutable fields; and $R1$ and $R2$, correlated arrays of features that are logically related.
Output: X', the adversarial NetFlow traffic sample.

$X' \leftarrow X$
$G \leftarrow getFGSMGrad(M, X, L)$
$max \leftarrow getMax(|G|)$
for $i = 1$ to $G.length$ **do**
  if $i \notin P$ and $max > 1$ **then**
    $X'[i] \leftarrow X'[i] + \frac{G[i]}{max}$
  else if $i \notin P$ and $max < 1$ **then**
    $X'[i] \leftarrow X'[i] + G[i]$
  end if
  if $X'[i] < 0$ **then**
    $X'[i] \leftarrow 0$
  else if $X'[i] > 1$ **then**
    $X'[i] \leftarrow 1$
  end if
  $X'[i] \leftarrow round(X'[i] * (V[i].max - V[i].min) + V[i].min)$
  $X'[i] \leftarrow \frac{X'[i] - V[i].min}{V[i].max - V[i].min}$
end for
for $j = 1$ to $R1.length$ **do**
  if $X'[R1[0]] < X'[R2[0]]$ **then**
    $X'[R1[0]] \leftarrow X'[R2[0]]$
  end if
end for
return X'

Table 1 The features and descriptions given for the NF-ToN-IoT-v2 dataset with the immutable features highlighted in GREY.

Feature	Description
L4_SRC_PORT	IPv4 source port number
L4_DST_PORT	IPv4 destination port number
PROTOCOL	IP protocol identifier byte
L7_PROTO	Layer 7 protocol (numeric)
IN_BYTES	Incoming number of bytes
OUT_BYTES	Outgoing number of bytes
IN_PKTS	Incoming number of packets
OUT_PKTS	Outgoing number of packets
FLOW_DURATION_MILLISECONDS	Flow duration in milliseconds
TCP_FLAGS	Cumulative of all TCP flags
CLIENT_TCP_FLAGS	Cumulative of all client TCP flags
SERVER_TCP_FLAGS	Cumulative of all server TCP flags
DURATION_IN	Client to Server stream duration (msec)
DURATION_OUT	Server to Client stream duration (msec)
MIN_TTL	Min flow TTL
MAX_TTL	Max flow TTL
LONGEST_FLOW_PKT	Longest packet (bytes) of the flow
SHORTEST_FLOW_PKT	Shortest packet (bytes) of the flow
MIN_IP_PKT_LEN	Len of the smallest flow IP packet observed
MAX_IP_PKT_LEN	Len of the largest flow IP packet observed
SRC_TO_DST_SECOND_BYTES	Src to dst Bytes/sec
DST_TO_SRC_SECOND_BYTES	Dst to src Bytes/sec
RETRANSMITTED_IN_BYTES	Number of retransmitted TCP flow bytes (src->dst)
RETRANSMITTED_IN_PKTS	Number of retransmitted TCP packets (src->dst)
RETRANSMITTED_OUT_BYTES	Number of retransmitted TCP flow bytes (dst->src)
RETRANSMITTED_OUT_PKTS	Number of retransmitted TCP flow packets (dst->src)
SRC_TO_DST_AVG_THROUGHPUT	Src to dst average thpt (bps)
DST_TO_SRC_AVG_THROUGHPUT	Dst to src average thpt (bps)
NUM_PKTS_UP_TO_128_BYTES	Packets whose IP size <= 128
NUM_PKTS_128_TO_256_BYTES	Packets whose IP size > 128 and <= 256
NUM_PKTS_256_TO_512_BYTES	Packets whose IP size > 256 and <= 512
NUM_PKTS_512_TO_1024_BYTES	Packets whose IP size > 512 and <= 1024
NUM_PKTS_1024_TO_1514_BYTES	Packets whose IP size > 1024 and <= 1514
TCP_WIN_MAX_IN	Max TCP Window (src->dst)
TCP_WIN_MAX_OUT	Max TCP Window (dst->src)
ICMP_TYPE	ICMP Type * 256 + ICMP code
ICMP_IPV4_TYPE	ICMP Type
DNS_QUERY_ID	DNS query transaction Id
DNS_QUERY_TYPE	DNS query type (e.g., 1=A, 2=NS)
DNS_TTL_ANSWER	TTL of the first A record (if any)
FTP_COMMAND_RETCODE	FTP client command return code

(FGSM) [4], a white-box attack that uses the sign of the gradients of the cost function of a classifier with respect to an input x to craft an adversarial example x':

$$x' = x + \epsilon * sign(\nabla_x J(\theta, x, y)),$$

where x is the input of IoT traffic, x' is the adversarial example, ϵ is an arbitrary constant of multiplication to meter the effects of the perturbation, θ represents the parameters of the model, y is the expected label for x, i.e., types of malicious traffic, and $\nabla_x J(\theta, x, y)$ retrieves the gradients of the cost function with respect to x. We choose FGSM because it is generally considered efficient and effective against neural networks such as CNN-based classifiers.

The adversarial perturbations in this research were generated using a variant of the FGSM perturbation process. Under the normal FGSM process, the values of the gradient of the trained model's cost function with respect to the input x are quickly calculated using backpropagation. The signs of the gradients (-1 for negative values, 1 for positive values, and 0 for 0) are then multiplied by a constant ϵ (which is usually less than 1) and piecewise added to x to produce the adversarial example x'. However, this approach applies $+/-\epsilon$ (or 0) evenly to all elements of the input x, regardless of the magnitudes of the original values of the gradient. This obviously results in a loss of scale and sensitivity for the perturbations. Moreover, this approach is ill-suited for the normalized NetFlow traffic data used in this research, as the value of ϵ in one normalized NetFlow feature may be orders of magnitude greater than the value of ϵ for a different feature once these features are converted back to their original scales (i.e., the normalization is reversed).

Our novel adaptive adversarial IoT network traffic generation algorithm is given formally as Algorithm 1. As a rule developed in this research, the perturbations generated must be scaled properly and their magnitudes restricted to

the range [0,1], as any perturbations beyond that range are nonsensical to the input. Unlike previous work, our perturbations to the mutable fields are not restricted to only be positive, as the goal of this research is to intelligently generate templates of what adversarial traffic that deceives a traffic classifier *could* look like according to our machine learning model.

Since the magnitudes of the values of the gradient are typically much less than 1, it makes sense to simply add the gradient itself – rather than the signs of the elements of the gradient – to the input x to create x', as this typically results in small and responsive perturbations to the input. However, an analysis of generated gradients for samples in the test traffic data reveals that in some cases, the magnitudes of certain values of the gradients exceeds 1, thus making these perturbations unreasonable. An obvious solution to this problem is therefore to adaptively scale these gradient-based perturbations to ensure that the largest value in magnitude of the perturbations is less than or equal to 1. The adaptive FGSM is promising for generating finer-grained, adaptively scaled perturbations in this research. While this approach does not guarantee that the resulting perturbations are minimal, it does make the perturbations at the very least reasonable by preventing these perturbation elements from exceeding the maximum values possible for the normalized features of IoT network traffic. Once scaled to ensure that the magnitudes of all elements in the gradient conform to the range [0,1], these elements can then be added as normal to the input with several restrictions.

To ensure that the generated adversarial network traffic still contains legal values, after adding the perturbations to the input, each feature is coerced to the range [0,1]. Post-processing of the perturbed network traffic is also necessary to ensure that no logical inconsistencies arise. For example, if after perturbing the network traffic, the shortest packet (bytes) is greater than the longest packet (bytes) of the flow, the logically inconsistent values are simply updated to restore the traffic to a logical state (in this case, the longest would therefore be set to the value of the shortest). Finally, to ensure that the normalized fields actually correspond to possible network flow record values, each feature in the normalized record is scaled back to its original scale prior to normalization, the features are rounded to integers, and the values are then normalized again to the range [0,1].

3.2 Adversarial Perturbation Visualizations

While generating meaningful adverbial network traffic data in the hopes of generating false-negatives (i.e., classifying attacks as benign) in the model is challenging, understanding why those changes happened and why they changed the model's prediction is much more challenging. Understanding such correlations of adversarial data and black-box machine learning model can also contribute to the explainable artificial intelligence (XAI) in more general sense. Therefore, this research developed a web-based, interactive visualization tool to gain more insight into how and why these changes occurred. This tool is meant to be interactive to allow for greater experimentation on how manipulating fields of IoT network traffic independently affects traffic classification.

3.2.1 Overview

An overview of IoT VisPerNet tool is illustrated in Figure 2. When the visualization tool's web user interface (UI) first loads, on the top of tool, a user sees an array of buttons, each representing one of the different types of attacks of IoT network traffic (e.g., DDoS, Injection, Ransomware, Scanning, etc.). Once a button is selected, it turns green and begins to load the visualizations.

There are three major views. First, Adversarial Perturbation Magnitude Views implemented by gradient bar charts (immediately below the attack classification buttons) visualize the similarities/differences between original and perturbed feature values. Second, the Adversarial Perturbation Distribution Views implemented by statistical box plots (immediately below the Adversarial Perturbation Magnitude Views) visualize how the perturbations are distributed for each feature. Finally, on the right-hand side of the tool, there is the Adversarial Sample Model Prediction View enabled by a fillable form that lets the user select an arbitrary feature and modify it with the numbers from different network traffic samples. The adversarial samples will be tested against the model for real-time prediction. Detailed design and explanations of each view are provided in next sections.

3.2.2 Adversarial Perturbation Magnitude Views

The design choice of this view as part of the visualization tool was based on its ability to show evolutions of network data. In particular, the visualization tool utilizes a gradient bar chart to show the difference between the original attack traffic fields and their corresponding perturbed traffic fields. Each mutable feature is shown on the y-axis and the traffic number is shown on the x-axis. For scalability, the visualization will display at most 50 network flows at a time, and the user can cycle through batches of network traffic samples. Each box is color-coded based on the color spectrum shown in Fig 3. The max value is assigned to the red side and the min value is assigned to the violet side of the spectrum. The only exception to this is if there is no change at all to a field, in which case the cell will appear white. Color-coding each box like this allows the user to quickly see what values had the greatest and smallest differences, as well as see how many other network traffic samples had a similar change in feature values.

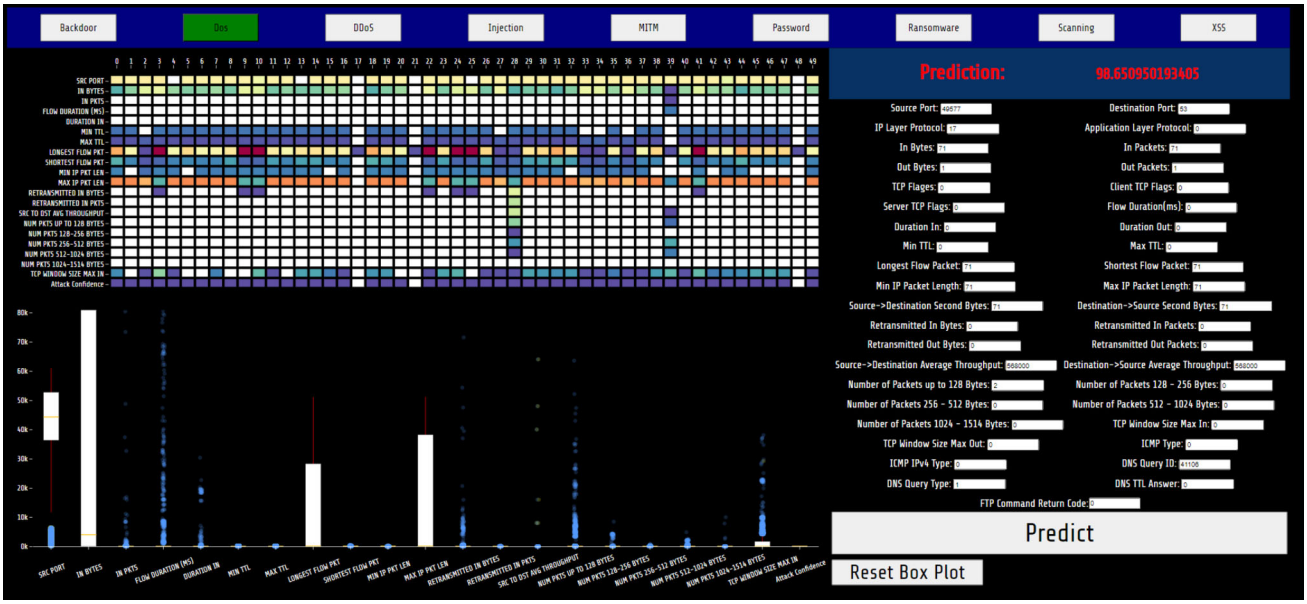

Fig. 2 The main view of interactive visualization tool IoT VisPerNet for adversarial perturbation analysis

Fig. 3 The color spectrum used for visualization (lower values on the cool side and higher values on the warm side)

To make this visualization even more useful, this custom visualization tool includes two features that make using this tool much simpler. The first is that when the user hovers their mouse over any of the boxes, it shows the original value, the perturbed value, the difference, and the max and min values for that feature (see Fig 4). The second feature is the ability to click on any of the numbers on the x-axis, which will automatically fill out the model prediction form with the original attack traffic values. Therefore, users can start modifying the values immediately without manually entering them. If the user clicks on the same number again, the form will be automatically filled with the perturbed values for that column. Both these features automate perturbation data analysis and allow for a better understanding of the data and a much more straightforward way of interacting with the visualization tool.

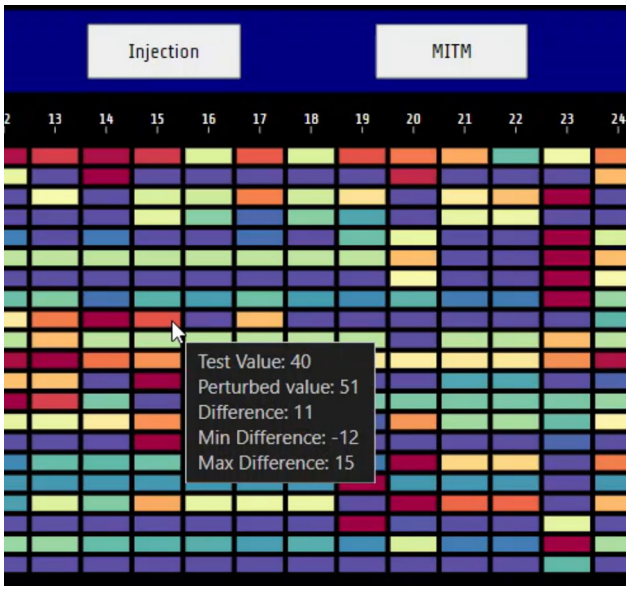

Fig. 4 A detail of a feature from the adversarial perturbation view made available by hovering over the feature. Clicking the numbers will automatically fill out the model prediction form

3.2.3 Adversarial Perturbation Distribution Views

The design choice of this view as part of the visualization tool was based on its ability to show network data distribution for each feature. In the visualization, all the mutable values are shown on the x-axis and the numbers for the current range of values are shown on the y-axis. The outliers are represented by the BLUE and YELLOW circles. If the user hovers their mouse over these outliers, the tool will display the value for the data point. The BLUE circles represent a value from a perturbed traffic flow while the YELLOW circles are original attack values. The upper and lower extreme values are shown by the RED line coming from the ends of the white rectangle. The WHITE rectangle is the boundary between the lower and upper quartiles, with the horizontal ORANGE line representing the median point. Hovering over this line will display the value of the median. Since each feature may have different scale, on the right side of the visualization, there is a "brushing" tool that will be seen once the user moves

their mouse over the area. Once the mouse turns into a plus sign, the user may then click and drag up or down to select the next range of values to be seen in detail. This is useful as some of the features have large values and some have small values, so this tool allows the user to adjust the scope of the visualization better. If the user accidentally selects a portion they did not want to see, they can click the "Reset Box Plot" button to bring the visualization back to the state it was at when it was first loaded.

3.2.4 Adversarial Sample Model Prediction View

The design choice of this view as part of the visualization tool was based on its ability to test hypotheses a user (admin/attacker) might have about the network traffic data. Specifically, if the user notices a trend that they think might be causing the model to make false predictions, the user can use the form to test this hypothesis using different values for their network traffic. The form contains every feature that is available in the network traffic, not just the mutable ones. Once all the text boxes are filled automaticlaly by clicking in the Adversarial Perturbation Magnitude View and modified accordingly from the insights derived from Adversarial Perturbation Distribution View (fields left blank will default to zero), the user can click the "Predict" button, which will pass the data to the machine learning classifier for a real-time prediction. The text above the form will then turn RED if the traffic is classified as attack traffic (with probabilities), GREEN if it is classified as benign, and YELLOW if it is classified with a confidence of 50% (which indicates model uncertainty).

4 Experiment, Evaluation and Discussion

In this section, we introduce the dataset used for the experimental study and system evaluation. We analyze the performance of adaptive perturbations and adversarial attacks on IoT network traffic classification. Through visual analytics, we provide insights into the patterns and correlations of perturbed features, emphasizing their significance in influencing model predictions.

4.1 Data and Processing

The data used in this research is the NF-ToN-IoT-v2 network traffic dataset, which distilled the heterogeneous IoT network traffic data simulated and captured in the testbed architecture [12, 16] into vectors containing 43 NetFlow version 9 features. The names and descriptions of these NetFlow record fields are shown in Table 1. Note that highlighted features indicate immutable features which were unchanged by the perturbation process. The NF-ToN-IoT-v2 dataset contains 16,940,496 flow records, 63.99% of which are attack traffic and 36.01% of which are benign traffic flows. The attack traffic flows consist of 9 different attack classes: Backdoor, Dos, DDoS, Injection, Man-in-the-Middle, Password, Ransomware, Scanning, and XSS attacks.

Table 2 Confusion Matrix for the 1D-CNN Traffic Classifier

		Predicted Label		
		Benign	Attack	Total
Actual Label	Benign	98195	1780	99975
	Attack	1120	98906	100026
	Total	99315	100686	200001

For the machine learning experiments, the dataset was randomized and balanced to contain 500,001 benign samples and 500,001 attack flows. All 1,000,002 samples were then shuffled and divided into training and testing sets according to an 80%-20% split. In addition to 99,315 benign samples, the test set contained 100,686 attack samples covering all nine attack classes. The training set was further split into training and validation sets using another 80%-20% split.

In every set, each feature for every record was normalized to a value in the range [0, 1] using min-max normalization, where the minimum and maximum values for each feature were either observed from extreme values for each feature in the 1,000,002 flows (as with features such as LONGEST_FLOW_PACKET) or calculated according to the inherent feature attributes (as with L4_SRC_PORT, which can have a maximum value of $2^{16} - 1$). Finally, the features for IPv4 addresses were trimmed from all records, as these values can potentially make ML models less robust by encouraging overfitting to match the IP signatures of benign and attack traffic.

4.2 CNN Classifier

The network traffic classification model chosen for our experimental study is an adaptation of deep 1D convolutional neural network (CNN). This model was implemented using the Keras deep learning API with a TensorFlow backend. The model was trained on the training data for 12 epochs to prevent overfitting, which was regulated by monitoring the model's performance on the validation data. The evaluation metric used was accuracy, with binary cross-entropy as the model's loss function, batch sizes of 64, and Adam as the optimizer. After training, the model achieved 98.55% accuracy on the test data. The confusion matrix (with a decision threshold of 0.5, where 0 is the label for benign traffic and 1 is for attack traffic) for the model performance on the test data is given in Table 2. This high model accuracy, precision, and recall indicate that this model is robust and therefore a good target for adversarial examples.

Table 3 Recall of the 1D-CNN Classifier with Respect to Test Attack Traffic Before and After Perturbations

	Predicted Benign	Predicted Attack
Before Perturbations	1120	98906
After Perturbations	72835	27191

4.3 Adversarial Attack Results

During the perturbation process, all 100,026 attack vectors from the test set were perturbed using a single round of the perturbation algorithm discussed in Section 3.1. Following this perturbation process, each perturbed attack sample was converted back into NetFlow form and then normalized again to ensure that the resulting perturbed traffic corresponded to NetFlow traffic. As this research is only interested in creating adversarial attack traffic samples, no benign traffic samples were perturbed. The decision threshold was set to 0.5 for binary classification task, meaning that perturbed attack samples for which the model gave a prediction less than this threshold were classified as benign traffic, and otherwise this traffic was classified as attack traffic. As Table 3 shows, the perturbations from this research caused over a 6,400% increase in the number of false negatives generated for the test attack traffic by the model. In terms of classification measurements, these perturbations decreased the recall of the model with respect to attack traffic from 98.9% to 27.2%, a dramatic 71.7% drop in recall.

4.4 Adversarial Perturbation Insights

It is insightful to observe that some features of the IoT network traffic were perturbed more heavily than others across all attack samples. The average magnitudes and their unnormalized values for all mutable NetFlow fields are shown in Table 4. In particular, the field MAX_IP_PKT_LEN experienced the largest average changes in terms of magnitude, with the average magnitude of perturbations applied to this field being 0.1882, which equates to the unnormalized value of 9,649.2. Conversely, the L4_SRC_PORT field was subject to the smallest average changes in terms of magnitude, with the average magnitude for the perturbations on this feature being 0.0004 and the related unnormalized value being 28.1.For every mutable feature, positive perturbation magnitude values correlate to an increase in the feature's value. It is important to note that the perturbations analyzed here are the effectual perturbations, rather than the raw perturbation values. Specifically, these perturbations correspond to the normalized changes in value for the NetFlow fields after each traffic sample was perturbed, not the exact perturbation values retrieved through the perturbation process.

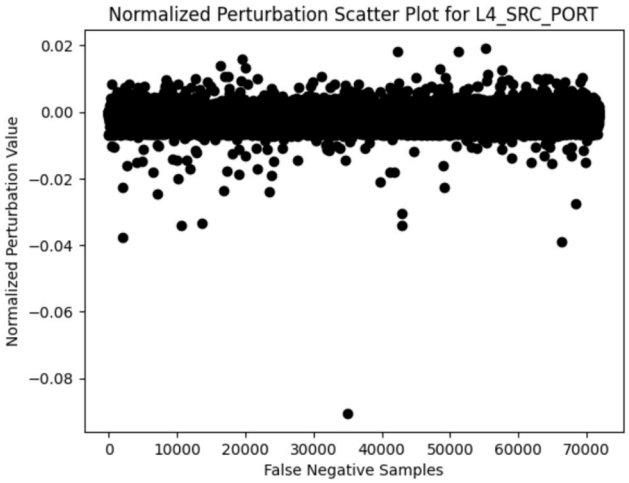

Fig. 5 The perturbation values (dominantly small changes around 0) for the L4_SRC_PORT field of successfully misclassified adversarial samples

The majority of all perturbations were positive, indicating that one method attackers can use for generating adversarial attack traffic that can fool a machine learning classifier is simply to induce increases to all the values of the mutable NetFlow fields of the traffic. However, this approach is naïve and would likely result in either suspicious traffic or traffic that could no longer be sent. It is therefore necessary to look at the gradient-based perturbations developed in this research for insight into how more responsive changes to certain NetFlow fields contribute to traffic misclassification.

In reviewing the effective perturbations (i.e., ones that resulted in traffic misclassification), it was evident that the perturbations for some fields, such as L4_SRC_PORT, RETRANSMITTED_IN_BYTES, and RETRANSMITTED_IN_PKTS, were highly concentrated around 0. Figure 5 shows the perturbation scatter plot for the L4_SRC_PORT feature. The distribution of perturbations suggests that regardless of the attack traffic, minimal to no perturbations of these fields were necessary to affect the traffic's classification by the model. This hypothesis is supported by the experiment data in Table 5 (see later discussion). This can mean that either these features are not so important when it comes to traffic misclassification or, conversely, that small changes to these features induce large classification errors. Regardless, this information indicates that attackers should focus on small changes to these values and that security managers should be aware of the effects of these minor adjustments.

On the other hand, perturbations on the features IN_BYTES, IN_PKTS, NUM_PKTS_UP_TO_128_BYTES, and NUM_PKTS_256_TO_512_BYTES varied greatly in the positive direction, spanning almost the entire range from 0 to 1 (see Fig 6, which shows the perturbation distribution for

Table 4 Average Perturbation Magnitudes on Mutable Traffic Fields

Name	Avg. Magnitude	Avg. Magnitude Value
L4_SRC_PORT	0.0004	28.1
IN_BYTES	0.0819	2556419.4
IN_PKTS	0.1696	67717.4
FLOW_DURATION_MILLISECONDS	0.0082	35219.5
DURATION_IN	0.0207	2471.9
MIN_TTL	0.0049	1.2
MAX_TTL	0.0159	4.1
LONGEST_FLOW_PKT	0.1359	6971.7
SHORTEST_FLOW_PKT	0.0107	43.7
MIN_IP_PKT_LEN	0.0161	5.8
MAX_IP_PKT_LEN	0.1882	9649.2
RETRANSMITTED_IN_BYTES	0.0051	14062.8
RETRANSMITTED_IN_PKTS	0.0013	13.7
SRC_TO_DST_AVG_THROUGHPUT	0.0419	127322101.1
NUM_PKTS_UP_TO_128_BYTES	0.1177	47007.2
NUM_PKTS_128_TO_256_BYTES	0.0053	71.2
NUM_PKTS_256_TO_512_BYTES	0.0785	807.9
NUM_PKTS_512_TO_1024_BYTES	0.0459	282.4
NUM_PKTS_1024_TO_1514_BYTES	0.0092	183.5
TCP_WIN_MAX_IN	0.0120	783.7

IN_PKTS). This indicates that the necessary perturbations on these fields were much more dependent on the attack traffic under consideration but still usually required an increase in the values of these fields. Less can confidently be said about these perturbations and what they indicate except that their seemingly random uniform positive distribution demonstrates the necessity of having a gradient-based perturbation algorithm to systematically generate appropriately scaled perturbations.

Interestingly, several of the perturbation fields appeared to show distinct, nonuniform patterns. For example, the LONGEST_FLOW_PKT field shows perturbations clustered near the values of 0, 0.5, and 1, with relatively few perturbation values straying from these clusters (see Fig 7). Similar trends appear in the data for MAX_IP_PKT_LEN, SRC_TO_DST_AVG_THROUGHPUT, and NUM_PKTS_512_TO_1024_BYTES. This stratification in the scatter plots for these perturbations suggests that there exist distinct values that attackers can use to make adversarial attack traffic more deceptive to machine learning models depending on specific type of attack traffic under consideration.

Other trends in the perturbation data appeared. Specifically, the scatter plots for FLOW_DURATION_MILLISECONDS, DURATION_IN, MAX_TTL, SHORTEST_FLOW_PKT, NUM_PKTS_128_TO_256_BYTES, NUM_PKTS_1024_TO_1514_BYTES, and TCP_WIN_MAX_IN all appeared to show striking preferences for perturbations in the range [0,0.2]. Knowing this range of viable

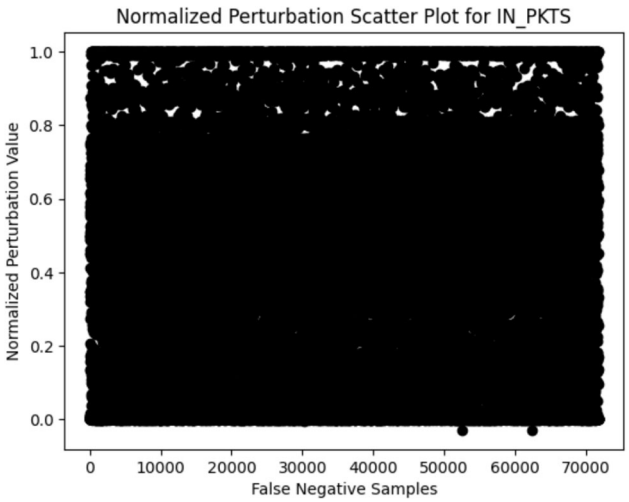

Fig. 6 The perturbation values (uniform positive distribution) for the IN_PKTS field of successfully misclassified samples

perturbations would be useful for both attackers creating adversarial attacks and defenders immunizing machine learning based IDS against these attacks.

It is also vital to analyze the high-level trends in how samples from different attack classes were perturbed, as attack traffic from the same classes are likely to receive similar perturbations. For an overview of these trends, Table 5 gives the median values of the perturbations in the perturbed fields of the different attack classes. Median is chosen rather than

Table 5 Median Values of Perturbations in Perturbed Fields of Different Attack Classes

	Backdoor	DoS	DDoS	Injection	MITM	Password	Ransomware	Scanning	XSS
SRC_PORT	0	3.05e-5	-1.53e-5	-1.53e-5	1.53e-5	-4.58e-5	-0.00069	0	3.05e-05
IN_BYTES	2.69e-6	0.24800	-5.39e-6	-7.05e-6	0.28436	-1.38e-5	-0.00014	-1.28e-6	0.19786
IN_PKTS	-2.00-5	0	0.03921	0.26394	0	0.23395	0.43331	4.50e-5	0
FLOW_DURATION_MS	1.40e-6	0	-4.68e-5	0	0	0	0.02197	0.00272	0
DURATION_IN	0	0	0.00044	0.00401	0	0.00163	0	0	0
MIN_TTL	0	0.00392	0	0	0.00392	0	0	0	0.00392
MAX_TTL	0	0.00392	0	0	0.00392	0	0	0	0.00392
LONGEST_FLOW_PKT	0.00777	0.55148	0.01501	0.01501	0.08360	0.00753	0.00730	0.01260	0.08941
SHORTEST_FLOW_PKT	0	0.00815	-0.00136	0.01664	0.00951	0.00543	-0.00815	0	0.00951
MIN_IP_PKT_LEN	0	0	0	0	0.00280	0	0	0.00280	0
MAX_IP_PKT_LEN	0.11166	0.74589	0.14503	0.11966	0.22519	0.10581	0.08325	0.02498	0.18383
RETRANSMITTED_IN_BYTES	0	0	0	0.00335	0.00013	0.00711	0.01682	0	0.00024
RETRANSMITTED_IN_PKTS	0	0	0	0	0.00049	0	0	0	0
SRC_TO_DST_AVG_THROUGHPUT	2.94e-7	0	0	0.00720	0	0	-0.00103	0	0
NUM_PKTS_UP_TO_128_BYTES	1.75e-05	0	0.00012	0.02483	0	0.01298	-1.75e-5	0	0
NUM_PKTS_128_TO_256_BYTES	0	0	0	0	0	0	0.38822	0	0
NUM_PKTS_256_TO_512_BYTES	0	0	0.01613	0.0760	0	0.04275	0	0.00117	0
NUM_PKTS_512_TO_1024_BYTES	0	0	0	0	0	0	0.11662	0	0
NUM_PKTS_1024_TO_1514_BYTES	0	0	0.00247	0.00232	0	0.00081	0	0.00030	0
TCP_WIN_MAX_IN	0	0.01085	-0.00021	0.00139	0.04816	-0.00019	-0.00085	6.10e-5	0

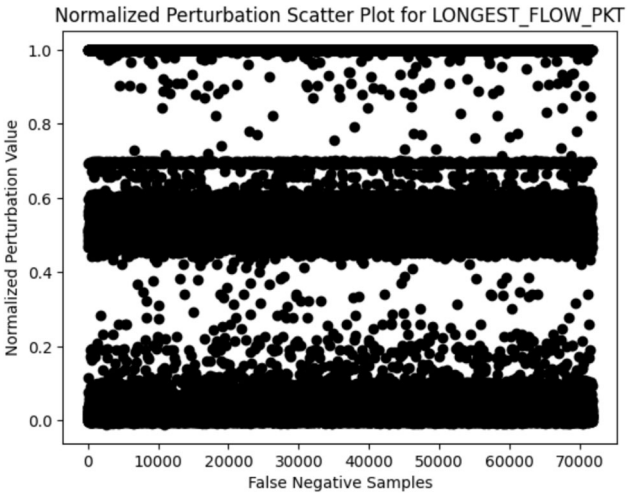

Fig. 7 The perturbation values (distinct, nonuniform distribution) for the LONGEST_FLOW_PKT field of successfully misclassified samples

mean as the measure of central tendency to decrease sensitivity to outliers in the perturbation data and give preference to clustered data. As can be seen in Table 5, the perturbations for certain attack classes appear to completely ignore some fields and focus on others. For example, for "xss" attacks (which represent roughly 23% of all attack samples in the test set), the perturbation algorithm appears to focus on extremely perturbing the IN_BYTES and MAX_IP_PKT_LEN fields while only marginally perturbing other fields or not applying any perturbations at all. Likewise, for "scanning" attacks (which represent about 35% of the test attack data), the perturbation algorithm appears to apply the largest perturbations to the LONGEST_FLOW_PKT field and barely -if any- perturbations to any other field. Perturbations are applied to a greater extent to attack classes such as the "password" attack class, which has one very sizeable median perturbation value of 0.43331 for the IN_PKTS field. However, even in these cases, the perturbation algorithm appears to ignore certain fields when perturbing specific attack classes. This suggests that by analyzing these perturbation signatures for the different attack classes, an attacker can fine-tune specific attacks to deceive a machine learning classifier. Security engineers can also use this information to create adversarial examples of deceptive attack traffic that can be used to improve the performance of classifiers on this adversarial traffic.

While these statistics for the overall perturbation values grant valuable insight into the high-level adjustments an adversary could use to manipulate the fields of IoT NetFlow traffic and create adversarial examples, they do not incorporate the interconnectedness of these fields. To gain a more complete understanding of how these fields changed in concert with one another, it is vital to look at representative examples of effective and ineffective perturbed attack traffic. Consider the following successfully perturbed adversarial attack sample given in Table 6, which was initially classified as attack traffic with a confidence of 0.9999937, but was then misclassified with a confidence of 0.0000043 after perturbations were applied. The entries in bold in the "Perturbed Value" column of Table 6 show which features of the NetFlow traffic changed. Only the mutable fields are shown for this sample for the sake of brevity. Interestingly, the dramatic drop in attack confidence by the classifier is due to both relatively small changes to fields such as source port, shortest packet flow, and retransmitted incoming packets *and* seemingly larger changes to fields such as average throughput and the maximum incoming TCP window. Also interesting is how certain mutable fields such as MIN_TTL were left unchanged, perhaps indicating that these fields are less significant for this specific attack sample. Overall, the perturbations from this sample seemed to target specific fields for changes, thus intelligently injecting enough noise into this sample to provoke a catastrophic misclassification. Clearly, our method of using a perturbation algorithm to create small, nuanced changes to IoT NetFlow traffic is superior to naïvely increasing the values for all fields and is therefore more realistic for attackers.

Despite this success, it is just as important to look at perturbed samples that failed to fool the classifier. Consider the sample given as Table 7, which was originally misclassified as a benign traffic sample with a confidence of 0.0033149 before being successfully classified as an attack sample after perturbations with a confidence of 0.9789484. While fewer fields were changed compared to the aforementioned successfully misclassified sample, the changes made to this sample appear to be much more drastic for several fields. For example, the field IN_PKTS was increased from 1 to 120,041, which would correspond to a massive increase in the number of packets sent by an attacker to a server and therefore would likely raise concerns. Further, some changes made by the perturbations, such as zeroing out the average throughput from the client to the server, would be inconsistent with real attack traffic. These mistakes show that the perturbation scheme appears to have limitations in some cases. However, the vast majority of successfully misclassified samples demonstrate that this method of generating adversarial attack templates is still statistically viable.

4.5 Adversarial Perturbation Visualization

This section also demonstrates a representative usage case by applying the IoT VisPerNet visualization tool to analyze the perturbed and unperturbed IoT network traffic for better understanding of how attack traffic may be manipulated to fool a machine learning classifier. This visualization offered insightful views to help identify vulnerabilities of the machine learning model that could be exploited by attackers.

Table 6 The Original and Perturbed IoT NetFlow Values for A Representative Successful Adversarial Attack Sample

Feature	Value	Perturbed Value
L4_SRC_PORT	861.0	**860.0**
IN_BYTES	4000.0	**4.0**
IN_PKTS	100.0	**407.0**
FLOW_DURATION_MILLISECONDS	0.0	0.0
DURATION_IN	0.0	**687.0**
MIN_TTL	71.0	71.0
MAX_TTL	71.0	71.0
LONGEST_FLOW_PKT	40.0	**479.0**
SHORTEST_FLOW_PKT	40.0	**41.0**
MIN_IP_PKT_LEN	0.0	0.0
MAX_IP_PKT_LEN	40.0	**28.0**
SRC_TO_DST_SECOND_BYTES	4000.0	4000.0
RETRANSMITTED_IN_BYTES	0.0	0.0
RETRANSMITTED_IN_PKTS	0.0	**5.0**
SRC_TO_DST_AVG_THROUGHPUT	32000000.0	**37577693.0**
NUM_PKTS_UP_TO_128_BYTES	100.0	**0.0**
NUM_PKTS_128_TO_256_BYTES	0.0	**167.0**
NUM_PKTS_256_TO_512_BYTES	0.0	0.0
NUM_PKTS_512_TO_1024_BYTES	0.0	**93.0**
NUM_PKTS_1024_TO_1514_BYTES	0.0	0.0
TCP_WIN_MAX_IN	74.0	**1722.0**
Attack Confidence	**0.9999937**	**0.0000043**

Table 7 The Original and Perturbed IoT NetFlow Values for an Unsuccessful Adversarial Attack Sample

Feature	Value	Perturbed Value
L4_SRC_PORT	48308.0	**48361.0**
IN_BYTES	105.0	**4.0**
IN_PKTS	1.0	**120041.0**
FLOW_DURATION_MILLISECONDS	0.0	0.0
DURATION_IN	0.0	0.0
MIN_TTL	0.0	0.0
MAX_TTL	0.0	**20.0**
LONGEST_FLOW_PKT	189.0	**3118.0**
SHORTEST_FLOW_PKT	105.0	**117.0**
MIN_IP_PKT_LEN	105.0	**108.0**
MAX_IP_PKT_LEN	189.0	**24305.0**
SRC_TO_DST_SECOND_BYTES	0.0	0.0
RETRANSMITTED_IN_BYTES	0.0	0.0
RETRANSMITTED_IN_PKTS	0.0	0.0
SRC_TO_DST_AVG_THROUGHPUT	840000.0	**0.0**
NUM_PKTS_UP_TO_128_BYTES	1.0	**0.0**
NUM_PKTS_128_TO_256_BYTES	1.0	**8439.0**
NUM_PKTS_256_TO_512_BYTES	0.0	0.0
NUM_PKTS_512_TO_1024_BYTES	0.0	0.0
NUM_PKTS_1024_TO_1514_BYTES	0.0	0.0
TCP_WIN_MAX_IN	0.0	**46.0**
Attack Confidence	**0.0033149**	**0.9789484**

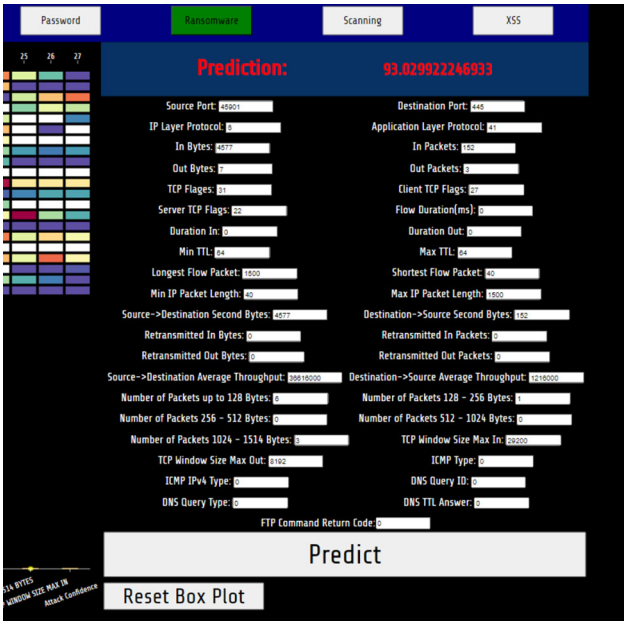

Fig. 8 Adversarial Sample Model Prediction View: an original ransomware flow with 95% confidence by the model as attack traffic

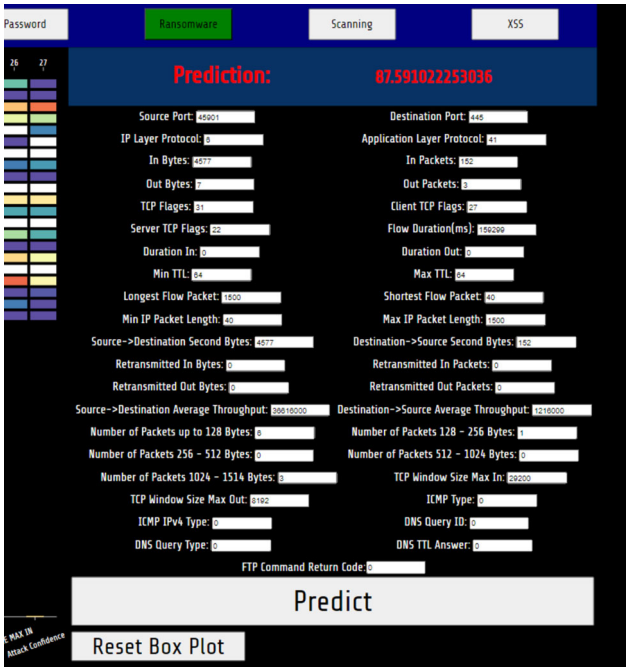

Fig. 9 Adversarial Sample Model Prediction View: model confidence level drops by changing one field "flow duration" to 159299

In particular, IoT VisPerNet laid bare trends in the values that the perturbation algorithm gave to certain features that caused the confidence of this perturbed traffic to change. Applying these values to attack data that had originally been classified as attack traffic with high confidence greatly lowered this confidence by the defending NIDS.

For example, IoT VisPerNet showed that the same value for flow duration had been chosen multiple times for perturbed ransomware traffic. To show that this perturbation value was indeed effective, this value for "flow duration" was applied to random ransomware traffic that was classified as an attack with 93% confidence before any perturbations were applied to it (see Fig 8). Just changing the flow duration to 159299 decreased the model's confidence by about 10%, as shown in Fig 9. However, once both the "flow duration" and the "duration in" fields (two logically related fields) were changed to 159299, this caused the model to predict the flow at around 20%, signifying that the real ransomware flow would now be classified as a "benign" flow (see Fig 10). Since not every value changed the model's prediction on the traffic this drastically, a bit of trial and error may be needed. However, being able to spot values that worked to mask a specific attack and tailor those values to sample traffic allowed for an increased chance of success when trying an adversarial attack on the machine learning model.

This interactive analysis results in findings of highly impactful fields in other types of attack traffic as well. For example, although almost all perturbations to "backdoor" attack traffic contained changes to the IN_BYTES, IN_PKTS, FLOW_DURATION_MILLISECONDS,

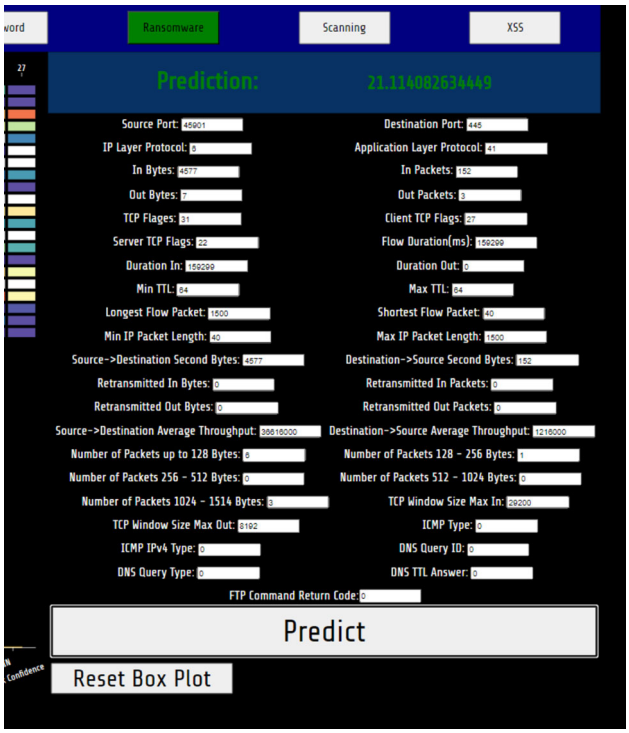

Fig. 10 Adversarial Sample Model Prediction View: further change in field "duration in" reduces the model prediction confidence to only 21%, rendering a successful IoT ransomware adversarial sample

LONGEST_FLOW_PKT, MAX_IP_PKT_LEN, RETRANSMITTED_IN_BYTES, SRC_TO_DST_AVG_THROUGHPUT, and NUM_PKTS_UP_TO_128_BYTES fields, this analysis found that only perturbations to the RETRANSMITTED_IN_BYTES field caused the classifier to drastically misclassify the perturbed "backdoor" attack traffic as "benign". The gradient bar chart for a selection of perturbed "backdoor" attacks is shown in Fig 11.

Other classes of attacks had fewer discernible trends in perturbation values. For example, although every field except for MIN_TTL and MAX_TTL were consistently perturbed for "password" attack traffic, it was difficult to tell if any one of these perturbed fields was the most impactful in terms of the change in the classifier's confidence on the perturbed traffic. However, the perturbation values for this attack class were largely consistent across different attack samples, as can be seen in Fig 12. Conversely, perturbations on "MITM" traffic held fewer consistent trends in the perturbation values overall, indicating that the perturbations in this case were much more sample-dependent and less indicative of the attack class. The adversarial perturbation magnitude view for "MITM" traffic can be seen in Fig 13.

A full list of observed trends in the perturbed fields of different attack classes by using the visualization tool is given as Table 8. Note that some attack classes, such as "injection" attacks, had no distinctive trends in perturbed fields. Note also that Table 5 closely mirrors Table 8 in that the non-zero entries in Table 5 mostly correspond to the "x" entries in Table 8. This provides evidences that the visual analysis of the data using IoT VisPerNet support the results from programmatic analysis of the perturbations, which demonstrates the effectiveness of IoT VisPerNet for unveiling trends in the IoT network data.

5 Conclusion, Limitation and Future Work

This research presented IoT VisPerNet, a tool for generating and analyzing the effects of gradient-based adversarial perturbations for adversarial machine learning attacks on IoT network traffic. The effects of the custom adversarial perturbation algorithm and the visual analytics from IoT VisPerNet show that the targeted machine learning network traffic classifier has several serious flaws that allow adversarial traffic to be misclassified. Importantly, the analysis highlights the ways in which an adversary could manipulate network traffic to fool machine learning based intrusion detection systems and so calls attention to these potential weak points in a NIDS.

IoT VisPerNet empowers users to visually identify key adversarial features by illustrating how they vary across different attack types. Network managers can leverage the tool to pinpoint the most perturbed features, quantify the extent of those perturbations, and uncover value ranges or feature combinations that lead to misclassifications. These insights provide a deeper understanding of the model's decision boundaries and enhance the security of real-world NIDS in IoT environments.

It is important to recognize the limitations of this research. Although powerful and informative, the perturbation algorithm resulted in changes to the NetFlow traffic that are not always consistent with real network traffic. Future research should also employ more intelligent guardrails to prevent perturbations from creating unrealistic network traffic. Overall, the process of isolating the significant fields of attack traffic is quite time-consuming, so being able to automate some of this identification in the future could improve perturbation analysis accuracy and efficiency. Finally, future research must investigate ways to preempt this sort of attack and design more robust artificial intelligence / machine learning based defense systems.

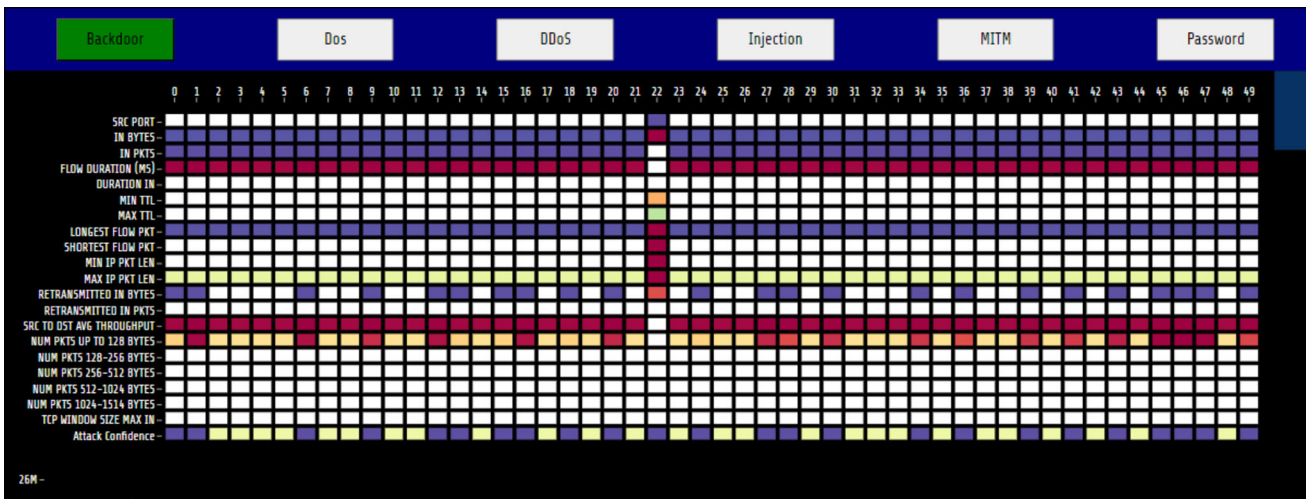

Fig. 11 Adversarial Perturbation Magnitude View conveys patterns for a selection of 50 perturbed "backdoor" adversarial attacks

Table 8 Observed Trends in Perturbed Fields of Different Attack Classes

	Backdoor	DoS	DDoS	Injection	MITM	Password	Ransomware	Scanning	XSS
SRC_PORT		x	x		x	x	x		x
IN_BYTES	x	x	x		x	x	x	x	x
IN_PKTS	x		x			x	x	x	
FLOW_DURATION_MS	x		x			x	x	x	
DURATION_IN			x			x			
MIN_TTL		x			x				x
MAX_TTL		x			x				x
LONGEST_FLOW_PKT	x	x	x		x	x	x	x	x
SHORTEST_FLOW_PKT		x	x		x	x	x		x
MIN_IP_PKT_LEN		x			x	x		x	x
MAX_IP_PKT_LEN	x	x	x		x	x	x	x	x
RETRANSMITTED_IN_BYTES	x					x	x		x
RETRANSMITTED_IN_PKTS						x			
SRC_TO_DST_AVG_THROUGHPUT	x					x	x		
NUM_PKTS_UP_TO_128_BYTES	x		x			x	x		
NUM_PKTS_128_TO_256_BYTES							x		
NUM_PKTS_256_TO_512_BYTES			x			x		x	
NUM_PKTS_512_TO_1024_BYTES						x	x		
NUM_PKTS_1024_TO_1514_BYTES			x			x	x	x	
TCP_WIN_MAX_IN		x	x		x	x	x	x	x

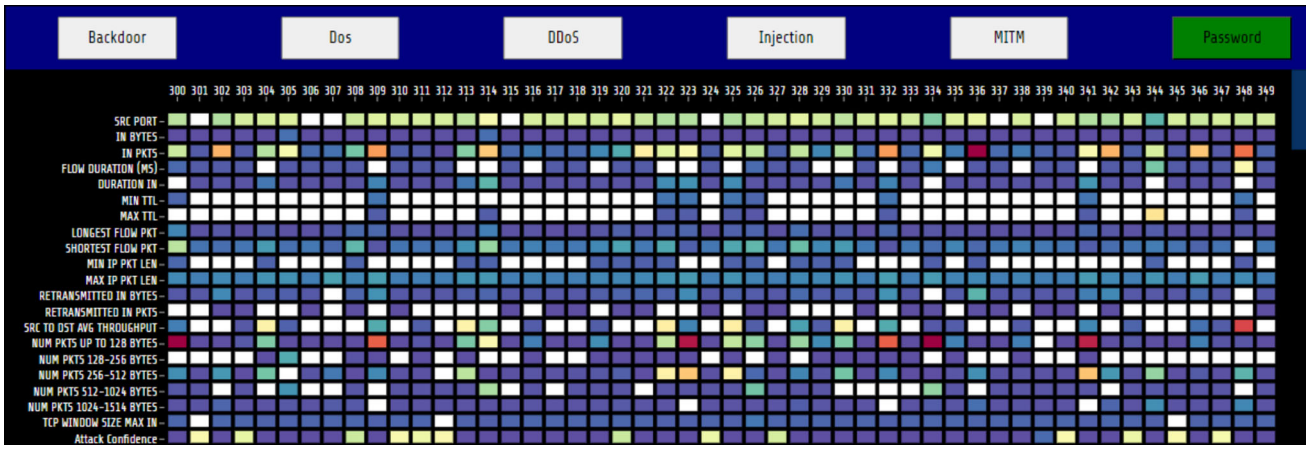

Fig. 12 Adversarial Perturbation Magnitude View conveys patterns for a selection of 50 perturbed "password" adversarial attacks

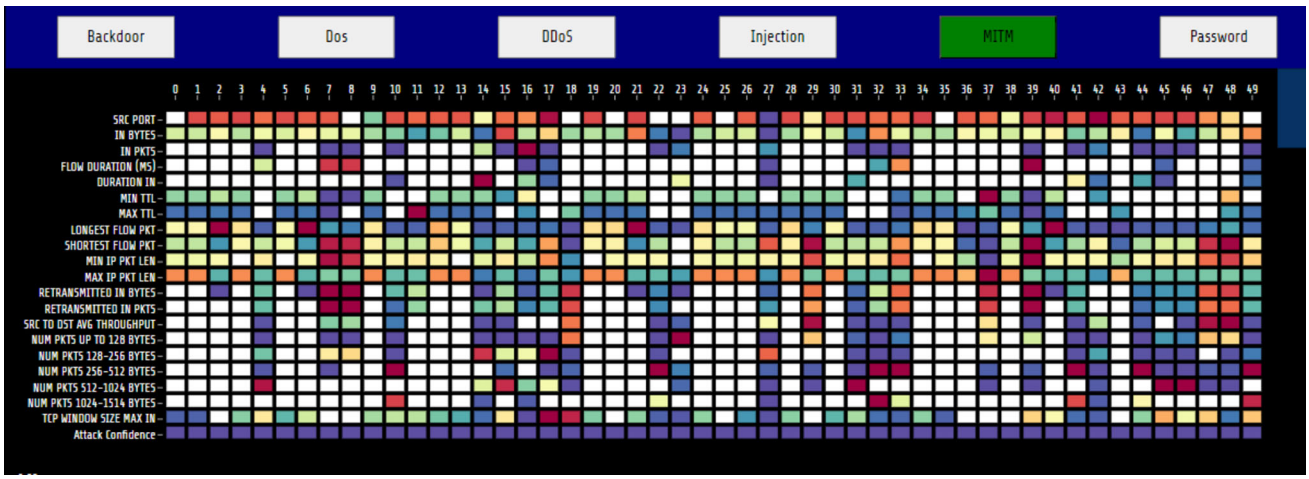

Fig. 13 Adversarial Perturbation Magnitude View conveys patterns for a selection of 50 perturbed "MITM" adversarial attacks

6 Compliance with Ethical Standards

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors. Conflict of Interest: Authors declare that they have no conflict of interest.

References

1. Alhajjar, E., Maxwell, P., Bastian, N.: Adversarial machine learning in network intrusion detection systems. Expert Syst. Appl. **186**(115), 782 (2021)
2. Cloudflare: What is the Mirai Botnet? https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/, [Online; Accessed 22-January-2025] (2025)
3. Godinho, P.I.A., Meiguins, B.S., Meiguins, A.S.G., Casseb do Carmo R.M., de Brito Garcia, M., Almeida, L.H., Lourenco, R.: Prisma - a multidimensional information visualization tool using multiple coordinated views. In: 11th International Conference Information Visualization (IV), pp. 23–32. Zurich, Switzerland (2007)
4. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations, San Deigo, CA (2015)
5. Gregory, J., Liao, Q.: Adversarial spam generation using adaptive gradient-based word embedding perturbations. In: IEEE International Conference on Artificial Intelligence, Blockchain, and Internet of Things (AIBThings), pp. 1–5. Mt. Pleasant, MI (2023)
6. Hu, Y., Tian, J., Ma, J., Ying, J.: A novel way to generate adversarial network traffic samples against network traffic classification. Wireless Communications and Mobile Computting (2021)
7. Kim, T.Y., Cho, S.B.: Web traffic anomaly detection using c-lstm neural networks. Expert Syst. Appl. **106**, 66–76 (2018)
8. Long, Z., Jinsong, W.: Network Traffic Classification Based On A Deep Learning Approach Using NetFlow Data. Comput. J. **66**(8), 1882–1892 (2022)
9. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., Lloret, J.: Network traffic classifier with convolutional and recurrent neural networks for internet of things. IEEE Access 5:18,042–18,050, (2017) https://doi.org/10.1109/ACCESS.2017.2747560
10. Mainuddin, M., Duan, Z., Dong, Y., Salman, S., Taami, T.: Iot device identification based on network traffic characteristics. In: IEEE Global Communications Conference (GLOBECOM), Rio de Janeiro, Brazil, pp 6067–6072, (2022) https://doi.org/10.1109/GLOBECOM48099.2022.10001639

11. Minh, D., Wang, H.X., Li, Y.F., Nguyen, T.N.: Explainable artificial intelligence: a comprehensive review. Artif. Intell. Rev. **55**, 3503–3568 (2022)
12. Moustafa, N.: A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. Sustain. Cities Soc. **72**(102), 994 (2021)
13. Qazi, E.U.H., Almorjan, A., Zia, T.: A one-dimensional convolutional neural network (1d-cnn) based deep learning system for network intrusion detection. Applied Sciences 12(16) (2022)
14. Ren, X., Gu, H., Wei, W.: Tree-rnn: Tree structural recurrent neural network for network traffic classification. Expert Syst. Appl. **167**(114), 363 (2021)
15. Santos, L., Gonçalves, R., Rabadão, C., Martins, J.: A flow-based intrusion detection framework for internet of things networks. Clust. Comput. **26**, 37–57 (2021)
16. Sarhan, M., Layeghy, S., Portmann, M.: Towards a standard feature set for network intrusion detection system datasets. Mobile Networks and Applications **27**(1), 357–370 (2022). https://doi.org/10.1007/s11036-021-01843-0. (https://doi.org/10.1007/s11036-021-01843-0)
17. Sevastjanova, R., Beck, F., Ell, B., Turkay, C., Henkin, R., Butt, M., Keim, D.A., El-Assady, M.: Going beyond visualization : Verbalization as complementary medium to explain machine learning models. In: IEEE VIS Workshop on Visualization for AI Explainability, Berlin, Germany (2018)
18. Sinha, S.: State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally. https://iot-analytics.com/number-connected-iot-devices/, [Online; Accessed 22-January-2025] (2024)
19. Siwakoti, Y.R., Bhurtel, M., Rawat, D.B., Oest, A., Johnson, R.C.: Advances in iot security: Vulnerabilities, enabled criminal services, attacks, and countermeasures. IEEE Internet of Things Journal 10(13):11,224–11,239, (2023) https://doi.org/10.1109/JIOT.2023.3252594
20. Sudar, K.M., Deepalakshmi, P., Singh, A., Srinivasu, P.N.: TFAD: TCP flooding attack detection in software-defined networking using proxy-based and machine learning-based mechanisms. Clust. Comput. **26**(2), 1461–1477 (2023)
21. Vignesh, V., Pavithra, D., Dinakaran, K., Thirumalai, C.: Data analysis using box and whisker plot for stationary shop analysis. In: International Conference on Trends in Electronics and Informatics (ICEI), pp. 1072–1076. Tirunelveli, India (2017)
22. Wang, W., Zhu, M., Zeng, X., Ye, X., Sheng, Y.: Malware traffic classification using convolutional neural network for representation learning. In: International Conference on Information Networking (ICOIN), Da Nang, Vietnam, pp 712–717, (2017) https://doi.org/10.1109/ICOIN.2017.7899588
23. Xu, Y., Hong, W., Li, X., Song, J.: Parallel dual visualization of multidimensional multivariate data. In: IEEE International Conference on Integration Technology, pp. 263–268. Shenzhen, China (2007)