ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
Received: 27 August 2025; Accepted: 31 March 2026

https://doi.org/10.17559/TV-20250827002928
Original scientific paper

Enhancing Machine Learning for Anomaly Detection and Classification Using Entropy-Based Dataset Enrichment

Igor FOSIĆ*, Drago ŽAGAR

Abstract: In machine learning and classification, entropy holds significant potential. This paper introduces a method to calculate Shannon entropy across all features within individual records in four IDS datasets: CSE-CIC-IDS2018, CIC-IDS2017, UNSW-NB15, and LUFlow. Each dataset is reshaped according to NetFlow, allowing for easy acquisition from actual network devices. A comparison of dataset versions with and without the entropy feature showed that the proposed entropy calculation method improves classification performance, even though the number of features was reduced compared to the original dataset. Enhanced classification and anomaly detection results are evident through improved AUC metrics and confusion matrix outcomes.

Keywords: anomaly detection; dataset enrichment; entropy; machine learning; NetFlow

1 INTRODUCTION

The primary task of intrusion detection is to accurately identify information about potential threats and irregularities within a communication network, which can range from a simple structure to a highly complex architecture featuring heterogeneous technologies and intricate designs. This process requires a systematic approach to data analysis to ensure both efficiency and detection accuracy. An aspect of this task involves filtering and removing redundant attributes, as well as reducing the dimensionality of input data. This approach helps isolate the most critical and relevant attributes needed for identifying security threats. In heterogeneous networks, where traffic information often encompasses large volumes of data with diverse attribute types, managing such information becomes exceptionally challenging [1]. Therefore, processes such as dimensionality reduction, redundancy elimination, and the extraction of key attribute sets are vital for effective intrusion detection. These processes enable resource optimization, improve data processing speeds, and enhance the accuracy of results, which is particularly important in modern network environments characterized by growing data volumes and increasingly sophisticated threats. By incorporating advanced data analysis methods, such as machine learning and artificial intelligence, the intrusion detection process can be further enhanced, enabling faster responses to security incidents and reducing the risk of harmful consequences.

The widespread application of anomaly detection systems, which rely on the collection and detailed analysis of network data, significantly contributes to a deeper understanding of network events, traffic patterns, and the distinctions between normal and suspicious traffic. These systems enable proactive monitoring of network activities, identification of potential threats, and enhancement of overall information system security. Modern anomaly detection techniques are primarily based on flow data analysis, offering a broader perspective on network activities [2]. Although considered less accurate compared to methods that analyze individual data packets, with the application of specific and innovative approaches these techniques can deliver fast and reliable results, making them ideal for environments where response speed is critical. The latest research trends focus on achieving finer granularity in systematic anomaly detection procedures. Emphasis is placed on advanced data preprocessing methods, including cleaning, filtering, and transforming data into formats suitable for further analysis. Additionally, there is an increasing focus on the integration of machine learning, which enables automated pattern recognition and system adaptation to new threats, significantly enhancing detection capabilities and responses to security incidents. The proposed process offers a methodology for measuring the unpredictability of incoming packets in network traffic, using entropy as a parameter. In this context, entropy represents a measure of uncertainty or randomness of a given dataset, expressing the probability of an event occurring relative to the total number of possible events. Higher entropy indicates a greater degree of unpredictability, while lower entropy suggests higher predictability and fewer variations in network traffic. During an attack, especially in cases such as DoS and DDoS attacks, entropy changes significantly, and these changes depend on various factors, specifically the header attributes of the packets being analyzed [3].

This paper contributes to the development of a process for attribute reduction in datasets aimed at anomaly detection based on data flow, tailored to the NetFlow structure, with data enrichment using entropy information. The goal of this approach is to improve the accuracy of network traffic classification in anomaly detection. The proposed process integrates entropy-based data preprocessing and their analysis using machine learning methods. The research focuses on dataset classification using supervised machine learning, as well as the analysis and enrichment of NetFlow data that has undergone entropy-based preprocessing. Numerous studies indicate that the application of entropy-based approaches in data preprocessing for anomaly detection aligns with the needs of enterprise environments, which are frequently targeted by botnets, data breaches, and DDoS attacks.

The practical implication of the proposed method lies in its ability to significantly reduce the number of features required for accurate classification. By selecting only a limited number of NetFlow compatible attributes and enhancing them with a calculated entropy value, the

approach avoids reliance on the large, often redundant and undocumented feature sets found in original datasets. This not only simplifies the data pipeline and improves computational efficiency but also ensures applicability in realworld monitoring systems where only basic flow data is typically available.

In this study, a prediction for binary classification (whether an attack exists or not) was first performed using the original features of each dataset: UNSW-NB15, CIC-IDS-2017, LuFlow, and CIC-IDS-2018. Then, binary classification was conducted using selected features aligned with the structure of Netflow records. Finally, binary classification prediction was performed on datasets augmented with an additional feature representing the entropy of each individual record within the dataset. The key contributions of this work include the utilization of four datasets - UNSW-NB15, CIC-IDS-2017, LuFlow, and CIC-IDS-2018 - for intrusion detection systems. Additionally, data preprocessing techniques were applied to ensure class balance, particularly for attack classes. Data augmentation was performed to evaluate the performance of the developed models using metrics such as Accuracy Score, F2, ROC AUC, Recall, and the Confusion Matrix. Finally, the Random Forest algorithm and various machine learning techniques were implemented on augmented versions of the utilized datasets.

The structure of the paper is as follows: Section 2 reviews related work on dataset enrichment for cybersecurity applications. Section 3 outlines the machine learning methods applied in this study and presents the results. Section 4 offers a discussion on the findings, while Section 5 concludes the paper.

2 RELATED WORK

Intrusion Detection Systems (IDS) play a crucial role in cybersecurity, and recent research has explored various machine learning and deep learning techniques to enhance their effectiveness. Data augmentation has emerged as a key strategy for improving model performance by balancing datasets and generating synthetic attack data. Machine learning algorithms can benefit from enriched features and synthetic data. New feature sets and traffic generation frameworks have further refined IDS performance by capturing intricate patterns in network traffic.

Study [4] examines the datasets UNSW-NB15, CIC-IDS-2017, FLNET2023, and 5G-NIDD, which are widely used in cybersecurity research despite a notable imbalance between normal network traffic and attack instances. The research evaluates the effectiveness of five deep learning architectures on augmented versions of these datasets for intrusion detection. The results indicate that complex and computationally intensive architectures are not necessary, as simpler convolutional neural network (CNN) based models, when combined with data augmentation, can achieve high accuracy. The analysis further confirms the ability of deep neural networks to identify relevant features and process complex datasets effectively.

Paper [5] presents the development and evaluation of a generic framework for designing a machine learning based Intrusion Detection System (IDS) specifically for SCADA systems in power grids. In the offline training process within this framework, data from different sources were utilized, including 15 different data files. The training dataset was then augmented and balanced by generating synthetic data using the CT-GAN technique. Three classical machine learning algorithms - Decision Tree, Random Forest, and Gradient Boosting - were evaluated and compared using both cross validation and holdout validation techniques to identify the most effective model for intrusion detection in SCADA power grid systems. The results showed that models trained on augmented data outperformed those trained solely on real data.

A recent approach [6] for addressing the low performance of IDS due to a lack of attack data involved using unsupervised models based on Generative Adversarial Networks (GANs). This paper proposes utilizing sequence based generative models, such as SeqGAN and Seq2Seq, to enhance the ADFA-LD dataset. Experimental results showed that performance was better when the model was trained with augmented data compared to training with only the original dataset. In this work, ADFA-LD was augmented using sequence based deep learning and evaluated through IDS. The SeqGAN model, which processes sequences by modifying the well-known GAN model, and Seq2Seq, which has been successfully used in translations and chatbots, were used for data generation. Specific models were found to be effective in learning IDS with augmented data and evaluating IDS performance. When the parameters were correctly tuned, by using a new model or better organizing the data inputs and outputs, the model was able to generate sufficiently good data for successful IDS training.

The paper [7] proposes an enhancement to flow based IDS by expanding existing features with a new set of features to improve detection accuracy. The flows consist of packets with similar attributes (e.g., the same source and destination IP address) and features that can differentiate between normal and malicious traffic, such as the symmetry of the source IP address and the symmetry of the entire flow. Experimental results showed significant improvement in IDS detection accuracy and a reduction in the false positive rate compared to existing systems. By combining enriched features with the basic ones, better detection accuracy was achieved, and the false positive rate was reduced. These improvements were confirmed through a T test, which showed a significant difference in accuracy between the selected and basic features, indicating the effectiveness of feature enrichment in improving classifier performance.

The paper [8] presents realistic datasets representing traffic for 7 common attacks and one for normal network traffic. Based on these datasets, it was evaluated how enriched features, combining raw data from the substation, can significantly improve intrusion detection performance. The results suggest that this could improve the F1 score for masquerade attacks. A traffic generation framework called ERENO was proposed, and public and realistic IEC–61850 datasets were generated. ERENO generates datasets with 69 traffic features, which include attributes extracted from GOOSE and SV protocols, as well as newly generated features that explore interprotocol correlation and the consistency of sequential messages. The proposed framework was validated using the J48 decision tree algorithm, with 7 attack classes and benign traffic. It was

shown that the enriched features enable the J48 classifier to classify these attacks with high accuracy, precision, recall, and F1 score metrics.

These studies highlight the effectiveness of data augmentation, feature enrichment, and generative models in improving IDS accuracy and reliability. They demonstrate that both deep learning and classical machine learning approaches benefit from well-structured datasets, emphasizing the importance of dataset quality in cybersecurity research.

3 MATERIALS AND METHODS

In this section, we have provided a detailed description of our vision for machine learning (ML) classification and performance of anomaly detection with various size of datasets.

3.1 Datasets

Data can be collected by capturing real network traffic within a known network infrastructure or by using publicly available datasets to address a specific problem. During the search for publicly available data to use in the machine learning model, four datasets created in recent years were selected. These chosen datasets were used with all features. Tab. 1 presents the key characteristics of the selected datasets. Each dataset is described by the number of features, the number of records (traffic flows), and its size in megabytes. Additionally, the total number of records is provided for each dataset.

The diversity of these datasets is revealed in terms of the number of features, the total number of records and their size. This variation is essential, as different network environments and traffic patterns can affect the performance of machine learning models. For example, datasets with a higher number of features may provide more granular insights but could also increase computational complexity. On the other hand, datasets with fewer features may require more sophisticated feature engineering to capture relevant patterns. Furthermore, the size and volume of records can influence the scalability of the models being developed, especially in large-scale network environments. The Tab. 1 provides essential information about the datasets, which is important for their further use in training, testing, and evaluating machine learning models in the context of cybersecurity applications.

Table 1 The comparison of traditional, hybrid and SDN network [12]

Name	Number of features	Number of records	Size / Mbytes
UNSW-NB15	49	2540044	598
LUFlow	16	1056062	106
CSE-CIC-IDS2018	84	7948748	4054
CIC-IDS2017	79	1149535	354

The UNSW-NB15 dataset consists of raw network packets (PCAP files, Packet Capture) generated using the IXIA PerfectStorm tool in the Cyber Range laboratory of the Australian Centre for Cyber Security. This tool was used to simulate real, contemporary normal activities and synthetic cyberattacks. The dataset includes nine attack scenarios: Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. A total of 49 features were generated using the Argus and Bro IDS tools, with two class labels for network traffic [9]. The dataset contains two labels: one represents a binary classification distinguishing normal traffic from anomalies, while the other specifies the type of anomaly (cyberattack) if the traffic is classified as an anomaly. The features are categorized into several types, including categorical, numerical (integer, decimal, and binary), and temporal features.

LUFlow is a dataset for detecting network attacks based on data flows, containing telemetry data on cyberattacks targeting decoys in the communication networks of Lancaster University. Anomalies in the data flow are automatically labelled through correlation with Cyber Threat Intelligence sources, enabling continuous capture, labelling, and publishing of telemetry data in this dataset. Flows that could not be determined as malicious but are not part of the normal telemetry profile are labelled as outlier flows and included in the dataset to encourage further analysis. This dataset also contains normal traffic and is continuously updated thanks to the automatic labelling mechanism, allowing for the detection of new attack patterns in real time [10-12].

CSE-CIC-IDS2018 dataset includes seven different attack scenarios: Brute force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and network infiltration attacks, which account for thirteen different types of cyberattacks. In the dataset, records are stored in the form of 80 features that were extracted and processed from the captured traffic. The records are organized by date, with raw data collected over a 10-day period, including network traffic (PCAP files) and event logs (Windows and Linux Ubuntu) for each individual device. The feature extraction process from raw data used the CICFlowMeterV3 tool, and the extracted features are stored as CSV files (Comma Separated Values) [13].

The CIC-IDS2017 dataset contains both benign traffic and the latest common attacks, closely resembling real world data (PCAPs). It includes the results of network traffic analysis using CICFlowMeter, with labelled flows based on the timestamp, source and destination IPs, source and destination ports, protocols, and attacks. The dataset consists of packets captured over a 5-day period in 8 separate sessions, which were released into 8 files, each corresponding to one session. For machine learning research, the data is preformatted into Comma Separated Values (.csv) files [14, 15]. The network traffic is classified into 15 classes, with one representing normal traffic and the remaining 14 corresponding to different types of attacks [16].

3.2 Data Preprocessing and Enrichment

When gathering data for various purposes, such as machine learning, data mining, or other applications, it is important to have methods in place to identify and extract the relevant data required for specific analytical goals. Each machine learning system has distinct requirements for how data should be represented, often necessitating the extraction and preparation of data to suit the intended analysis. The choice of data for analysis can significantly affect the models being trained, making data preparation a

critical and often the most challenging phase of many machine learning or data mining projects. Typically, data preparation involves a series of transformations that must be repeated multiple times [17]. Despite the availability of data processing technologies, each transformation can involve substantial manual effort, requiring considerable time and resources. As a result, data preparation is often seen as the most difficult and time intensive part of the data analysis workflow. Messy and unorganized data present significant barriers to effective analysis, with research indicating that up to 70% of time spent on data analysis is devoted to data cleaning [18]. Data cleaning involves identifying and correcting (or removing) corrupted or inaccurate records from datasets, tables, or databases. The preprocessing of the datasets involved several steps to ensure the data was clean and properly formatted for the machine learning models. Initially, the features that provided sequence numbers, IP addresses, or timestamp information related to the flow of network traffic were removed, as they were not relevant for the classification tasks. These features could introduce unnecessary complexity or overfitting, so their exclusion was necessary for the integrity of the data.

Since the datasets included categorical features, these needed to be converted into numerical representations, as machine learning algorithms typically require numerical input. The conversion was performed in a manner tailored to the specific characteristics of each dataset. Categorical features, such as protocol labels, were converted using a Label Encoder. This technique was preferred over other methods, like One hot encoding, as it does not increase the dimensionality of the dataset by creating additional binary features, which can sometimes lead to computational inefficiencies.

A typical feature in the datasets is the Label, which categorizes the type of network traffic. This can either be benign traffic, labelled as "normal," or traffic that represents an anomaly, such as a cyberattack. To make these labels compatible with machine learning algorithms, categorical values were mapped to binary values, with "normal" traffic being assigned a 0 and anomalies being assigned a 1.

Another common issue across many of the datasets was the presence of NaN (Not a Number) values. These are problematic because they cannot be processed or converted into numerical values. To handle this, rows containing NaN values were removed from the dataset. Fortunately, the number of such rows was relatively small compared to the total number of records, so their removal did not significantly affect the overall size or representativeness of the data. This was considered an acceptable part of the data preprocessing process.

For decision tree based models like the Random Forest used in anomaly detection normalization or feature scaling is typically unnecessary, as these models split data based on feature thresholds rather than distance metrics. Consequently, normalization has little to no impact on model performance, making preprocessing simpler compared to algorithms that rely on distance calculations, such as K-NN or SVM.

One of the most significant challenges with these datasets was the imbalance between normal and anomalous traffic. In most cases, the datasets contained far more instances of normal traffic than anomalies, which could lead to biased model training, where the classifier is overly biased toward predicting normal traffic. To mitigate this, the datasets were adjusted to create a balanced distribution of anomaly and normal traffic records. After analyzing all the datasets, it was observed that the UNSW-NB15 dataset contains the fewest records labelled as anomalies, specifically 321283 records. Therefore, a decision was made to standardize all datasets, with the aim of ensuring that each dataset contains 642566 records, maintaining a 50% ratio of anomalies to normal traffic. This balancing step is recommended because class imbalances can significantly impact the model's performance. When the dataset is not balanced, the machine learning model may struggle to accurately detect the minority class, leading to poor classification results.

The classification performance of data in different datasets cannot be compared if these datasets differ in the number of features. This is the main reason why it is important to reduce or standardize the number of features in each dataset before any analysis is conducted. In the real world, network traffic data is collected from various network devices. One of the most commonly used protocols for gathering network traffic information is NetFlow. NetFlow v9 provides a basic set of features, including: 'sport' (source port), 'dsport' (destination port), 'proto' (protocol), 'dur' (connection duration), 'sbytes' (bytes sent from the source device), 'dbytes' (bytes sent to the destination), 'Spkts' (packets sent from the source), and 'Dpkts' (packets sent to the destination).

In addition to these basic features, each dataset typically contains a 'Label' feature, which indicates the type of traffic (normal or anomaly), bringing the total number of features to 9. However, the selected datasets contain many more features derived through various statistical calculations, such as aggregated traffic data or network behavior. While these additional features are useful for in depth analysis, they can create significant variation between datasets, making it difficult to directly compare their performance.

By eliminating redundant or unnecessary features and limiting the number of features to the basic set, we create simplified versions (v1) of the datasets that allow for more comparable classification results. In this way, we reduce the complexity of the data and allow classification models to better compare the performance of different datasets. This feature standardization process also contributes to a more accurate and efficient comparison of classification performance, which is essential for further research and the implementation of machine learning models in the context of network traffic anomaly detection. Complete adaptation to the NetFlow structure (version v2 of datasets) was not possible for all datasets as not all of them contained the required features. Specifically, the 'src_port' and 'protocol' features were missing for the CIC-IDS2017 dataset. Due to these shortcomings, it was not possible to fully adapt this dataset to the NetFlow format. Although not all reduced datasets were identical, they were more uniform than the original datasets presented in Tab. 1. This adjustment significantly reduced variations among the datasets, making them more suitable for further analysis and classification performance comparisons.

The third version of each dataset is obtained by adding an entropy feature to the dataset with a reduced number of features. Shannon's entropy was used for this feature. Shannon's entropy is commonly used in anomaly detection as it quantifies uncertainty and imbalance in the data, making it sensitive to irregular patterns. It can detect shifts in data distribution, which often indicate anomalies. Its simplicity, scalability, and ability to recognize new or rare anomalies across different types of data make it an effective tool for anomaly detection in various applications.

Table 2 Datasets versions overview

Name	Number of features	Number of records	Remark
UNSW-NB15_v1	42	642566	
UNSW-NB15_v2	9	642566	Netflow structure
UNSW-NB15_v3	10	642566	Netflow structure with entropy feature
LUFlow_v1	12	642566	
LUFlow_v2	9	642566	Netflow structure
LUFlow_v3	10	642566	Netflow structure with entropy feature
CSE-CIC-IDS2018_v1	80	642566	
CSE-CIC-IDS2018_v2	8	642566	Netflow structure
CSE-CIC-IDS2018_v3	9	642566	Netflow structure with entropy feature
CIC-IDS2017_v1	79	642566	
CIC-IDS2017_v2	6	642566	Netflow structure without 'src_port' and 'prot'
CIC-IDS2017_v3	7	642566	Netflow structure with entropy feature without 'src_port' and 'prot'

In this study, Shannon's entropy was calculated for all the features of each data flow to enhance anomaly detection. The formula for calculating entropy, as defined in [19], is:

$$H(X) = -\Sigma P(x_i)\log_n P(x_i) \quad (1)$$

where X represents the set of qualitative variables, and P represents their corresponding probabilities. Calculating the entropy of all features for each data flow allows for a better understanding of the data's structure and disorder, which is important for more accurate anomaly detection. This process improves the model's sensitivity to unusual or unexplained patterns. The calculation of Shannon's entropy for all records was performed using the scipy.stats module and the Python programming language, enabling efficient processing and analysis of large datasets.

By calculating the entropy and adding the new feature to the datasets with a reduced number of features, similar to the NetFlow protocol, the data preprocessing process was completed, resulting in three versions of each dataset as shown in Tab. 2.

Tab. 2 provides an overview of multiple dataset versions, each containing the same number of records (642, 566) but varying in the number and type of features. The original versions (v1) have the highest feature counts, while subsequent versions (v2 and v3) focus on streamlined NetFlow compatible features, with v3 versions including an additional entropy feature. It is important to note that certain dataset versions, particularly from CIC-IDS2017, do not include specific features such as 'src_port' and 'prot', either by design or due to their absence in the original data. This structured variation allows for comparative analysis of feature sets related to NetFlow representation and entropy inclusion across different datasets.

3.3 Machine Learning Classification and Evaluation

A machine learning technique was applied for classification and anomaly detection across all four observed datasets, with two additional variations for each dataset - one with a reduced number of features and another with an added entropy feature. Each dataset was split in a 70/30 ratio for the training and testing phases within the machine learning process.

Tab. 3 shows the comprehensive general table summarizing the performance of eight machine learning classifiers - XGBoost, Random Forest, Bagging (Decision Tree), Gradient Boosting, AdaBoost, Extra Trees, K-Nearest Neighbors (K-NN), and LinearSVC (Calibrated) - across four datasets (CIC2017, UNSW-NB15, CIC2018, LUFlow), now including F2 score, ROC AUC (PR AUC), and Rank per dataset. An average rank column is also included for overall comparison.

Table 3 Comprehensive table summarizing the performance of eight machine learning classifiers across four datasets (v1)

Classifier	CIC-IDS2017/F2 / AUC / Rank	UNSW-NB15/F2 / AUC / Rank	CSE-CIC-IDS2018/F2 / AUC / Rank	LUFlow/F2 / AUC / Rank	Avg. Rank
XGBoost	0.9999 / 1.0000 / 1	0.9964 / 0.9997 / 4	1.0000 / 1.0000 / 2	0.9995 / 1.0000 / 4	2.75
RandomForest	0.9995 / 0.9999 / 3	0.9968 / 0.9996 / 2	0.9999 / 1.0000 / 4	0.9997 / 0.9998 / 1	2.5
Bagging (DT)	0.9996 / 0.9997 / 2	0.9945 / 0.9975 / 6	0.9999 / 1.0000 / 3	0.9996 / 0.9997 / 3	3.5
GradientBoosting	0.9975 / 1.0000 / 5	0.9972 / 0.9996 / 1	0.9998 / 1.0000 / 6	0.9993 / 1.0000 / 5	4.25
AdaBoost	0.9960 / 0.9999 / 7	0.9963 / 0.9996 / 5	1.0000 / 1.0000 / 1	0.9983 / 0.9999 / 7	5
Extra Trees	0.9993 / 0.9999 / 4	0.9966 / 0.9996 / 3	0.9998 / 1.0000 / 5	0.9997 / 0.9997 / 2	3.5
K-NN	0.9965 / 0.9972 / 6	0.9651 / 0.9776 / 7	0.9988 / 0.9983 / 7	0.9980 / 0.9992 / 6	6.5
LinearSVC (Calib.)	0.3907 / 0.7099 / 8	0.0000 / 0.5000 / 8	0.9057 / 0.9032 / 8	0.9012 / 0.9382 / 8	8

The same testing was performed on all subsets (v2) adapted to the NetFlow structure, and the results are consolidated in Tab. 4.

Drawing from the detailed evaluation presented in Tabs. 3 and 4, Random Forest proves to be the most dependable and consistently effective machine learning classifier across various network intrusion detection

datasets and data formats. Its strengths include consistently high rankings, excellent performance metrics, robustness across diverse datasets, and favourable comparison with other algorithms. Random Forest clearly distinguishes itself as the top overall classifier for network intrusion detection due to its reliable, highlevel performance across multiple datasets and key evaluation criteria, making it the preferred choice for realworld applications.

Table 4 Comprehensive table summarizing the performance of eight machine learning classifiers across four datasets (v2)

Classifier	CIC-IDS2017/F2 / AUC / Rank	UNSW-NB15/F2 / AUC / Rank	CSE-CIC-IDS2018/F2 / AUC / Rank	LUFlow/F2 / AUC / Rank	Avg. Rank
RandomForest	0.9944 / 0.9976 / 1	0.9947 / 0.9992 / 2	0.9993 / 0.9997 / 1	0.9997 / 0.9998 / 1	1.25
Bagging (DT)	0.9941 / 0.9961 / 2	0.9922 / 0.9966 / 4	0.9990 / 0.9994 / 3	0.9996 / 0.9998 / 3	3
Extra Trees	0.9940 / 0.9948 / 3	0.9943 / 0.9989 / 3	0.9993 / 0.9998 / 2	0.9997 / 0.9998 / 2	2.5
XGBoost	0.9937 / 0.9998 / 4	0.9964 / 0.9997 / 1	0.9989 / 1.0000 / 4	0.9993 / 1.0000 / 4	3.25
GradientBoosting	0.9892 / 0.9993 / 5	0.9879 / 0.9980 / 5	0.9980 / 1.0000 / 6	0.9990 / 1.0000 / 5	5.25
K-NN	0.9785 / 0.9860 / 6	0.9721 / 0.9844 / 6	0.9681 / 0.9667 / 7	0.9987 / 0.9991 / 6	6.25
AdaBoost	0.9620 / 0.9973 / 7	0.9661 / 0.9961 / 7	0.9983 / 0.9999 / 5	0.9979 / 0.9999 / 7	6.5
LinearSVC (Calib.)	0.0000 / 0.5000 / 8	0.8814 / 0.7715 / 8	0.0000 / 0.5000 / 8	0.9300 / 0.9304 / 8	8

The Random Forest algorithm was employed for test data classification, consisting of multiple decision trees. Instead of relying on a single decision tree, this classifier aggregates predictions from each tree and determines the final classification outcome based on majority voting [20, 21]. The performance of the machine learning algorithm is assessed using metric values. The challenge of evaluating a classifier (i.e., measuring its quality) is addressed through a unified assessment that attempts to summarize specific conditions and considerations in the evaluation process [22].

The classification performance was evaluated using the F2 score, ROC-AUC, Recall, and the confusion matrix. If False Negative predictions are particularly critical, the combination of Recall, F2 score, and ROC-AUC enables a comprehensive analysis of the model, with a strong emphasis on reducing the number of missed positive cases. While Recall directly measures the model's ability to identify all actual positive instances, F2 score further emphasizes Recall over Precision, ensuring that the model prioritizes recognizing positive cases, even at the expense of a higher false positive rate. At the same time, ROC-AUC provides a broader view of the model's discriminative ability, allowing for the selection of an optimal classification threshold that can further minimize the occurrence of False Negative cases based on specific application requirements [23]. This evaluation facilitated an assessment of classification effectiveness and anomaly detection capability, offering valuable insights into the algorithm's performance and its applicability across different scenarios.

The Python script integrates all essential steps of data processing and classification analysis, including dataset loading, preprocessing, the application of machine learning techniques for classification, and the evaluation of classification performance using relevant metrics. In the first step, the data is loaded and undergoes basic analysis to identify potential inconsistencies or missing values. This is followed by preprocessing, which includes handling missing values and dimensionality reduction, depending on the characteristics of the dataset. After preprocessing, the data is split into training and test sets in a predefined ratio 70/30. The Random Forest machine learning algorithm is then applied to the training set to build the classification model. Once the model is trained, the test data is used to assess its accuracy and generalization ability. The classification performance is evaluated using several metric indicators, including Recall, F2 score, ROC-AUC, and the confusion matrix, ensuring a comprehensive analysis of the model's performance, with a particular emphasis on reducing False Negative cases in the context of anomaly detection. The entire process is implemented in the Python programming language, and the developed Python script is presented below in the form of pseudocode, labelled as Algorithm 1.

Algorithm 1:
Dataset enrichment with entropy feature and classification
    Data: Dataset (v1, v2, v3)
    Input: Dataset features
    Output1: Entropy feature
    Output2: Accuracy metrics
Import dataset
    Clean NaN values
    Clean unnecessary features
Dataset preprocessing
    Convert categorical features to numerical
    Adjust number of records
    For v2 and v3 dataset version
        Adjust number of features to Netflow structure
        For v3 dataset version
            for each row ∈ Data **do**
                for each value ∈ Input **do**
                    Determine feature value occurrence
                    Compute feature probability
                end
            Output 1: Shannon's row entropy
            end
Machine learning
    Split dataset into training and test subsets
    Train the model using the Random Forest classifie
    Predict and classify test data
Calculate accuracy metrics
    Output 2: confusion matrix, F2 score, ROC-AUC, Recall

This algorithm systematically prepares the dataset, enriches it with an entropy-based feature, applies a machine learning classification model, and evaluates its performance using relevant metrics. It ensures that data preprocessing aligns with the necessary structure before training a model, ultimately enhancing classification accuracy and anomaly detection.

Including entropy as a feature introduces a moderate computational overhead, as it requires calculating probabilities for each feature and computing entropy row-by-row. In timing tests, entropy calculation took ~25-27 seconds per dataset shown in Tab. 5, which is a significant

portion of the total script time. However, this overhead is manageable, and the added feature improves classification performance by providing additional discriminatory power - especially for detecting anomalies. Entropy also makes the pipeline more complex, as it requires column wise value distributions and is not natively available in many datasets. Nevertheless, with proper preprocessing, it can be a valuable feature in intrusion detection systems. In proposed implementation, the entropy calculation was applied to NetFlow adapted datasets (v2). This step was integrated after the data preprocessing pipeline and did not cause a significant increase in CPU and memory usage.

Table 5 Summary table of the timing results for all four datasets

Dataset	Entropy Calculation s	Classification Time / s	Total Script Time / s
CIC-IDS2017	25.3	2.62	42.02
CSE-CIC-IDS2018	27.3	2.92	159.79
LUFlow	26.96	3.18	31.78
UNSW-NB15	26.63	4.08	33.79

Table 6 Summary table of the timing results for all four datasets

Dataset	Features	AccuracyScore	TN	FP	FN	TP	Recall	F2 score	Precision-Recall AUC
UNSW-NB15									
Original (v1)	42	0.99352	95239	1146	103	96282	0.99893	0.99677	0.99961
NetFlow (v2)	9	0.99242	95282	1103	359	96026	0.99628	0.99474	0.99923
Entropy (v3)	10	0.99252	95260	1125	317	96068	0.99671	0.99504	0.99937
LUFlow									
Original (v1)	12	0.99968	96351	34	27	96358	0.99972	0.99971	0.99982
NetFlow (v2)	9	0.99969	96354	31	28	96357	0.99971	0.9997	0.99982
Entropy (v3)	10	0.99977	96356	29	16	96369	0.99983	0.99981	0.99992
CIC-IDS2017									
Original (v1)	79	0.99954	96339	46	43	96342	0.99955	0.99955	0.99991
NetFlow (v2)	6	0.99568	95907	478	355	96030	0.99632	0.99606	0.99899
Entropy (v3)	7	0.99578	95904	481	333	96052	0.99655	0.99624	0.99909
CSE-CIC-IDS2018									
Original (v1)	80	0.99992	96377	8	8	96377	0.99992	0.99992	0.99999
NetFlow (v2)	9	0.99915	96278	107	56	96329	0.99942	0.99931	0.99973
Entropy (v3)	10	0.99904	96269	116	70	96315	0.99927	0.99918	0.99977

Tab. 6 presents a comparison of various datasets (UNSW-NB15, LuFlow, CIC-IDS2017, and CSE-CIC-IDS2018) across different feature sets (Original (v1), NetFlow (v2), and Entropy (v3)), along with key classification metrics such as Accuracy Score, True Negatives (TN), False Positives (FP), False Negatives (FN), True Positives (TP), Recall, F2 score, and Precision-Recall AUC.

The analysis presented in Tab. 6 further confirms that the original feature sets consistently achieve the lowest FN values, underscoring their strong anomaly detection capabilities. While NF and NF Entropy provide the advantage of feature reduction, this benefit comes at the cost of increased FN, particularly in datasets such as UNSWB-15 and CIC-IDS2017. However, in LUFlow and CSE-CIC-IDS2018, the FN differences between the original and NF Entropy models are minimal, making feature reduction a viable alternative in these cases. The entropy-based models (v3 versions) demonstrate robust performance, exhibiting competitive or even superior False Negative (FN) and Precision-Recall AUC values compared to the NetFlow (v2) models. This suggests that the Entropy (v3) models can maintain high classification efficacy with a more compact feature set, relative to the size and complexity of features present in the original datasets (v1). This characteristic makes them a compelling choice in scenarios where feature reduction is a priority, without significantly compromising model performance. The only exception is the CSE-CIC-IDS2018 dataset, where the Entropy (v3) model exhibited slightly worse FN and PR-AUC results compared to its performance in other datasets. Despite the observed increase in FN in some cases, Precision-Recall AUC remains consistently high across all models, indicating that overall classification performance is not significantly degraded. Moreover, NF Entropy models frequently outperforms NF models, positioning it as the preferred reduced feature set across most datasets. These findings highlight the importance of carefully evaluating feature reduction methods to balance the trade-off between complexity reduction and maintaining optimal anomaly detection performance.

3.4 Impact of Features on Model Performance

Random Forest (RF) is an ensemble learning classification algorithm that constructs multiple decision trees independently and combines their outputs to improve prediction accuracy. These decision trees consist of internal and leaf nodes, where the internal nodes use selected features to make decisions, splitting the dataset into two subsets with similar responses. RF employs the bagging method and random feature selection to generate an uncorrelated forest of decision trees, which enhances predictive performance compared to a single tree. Bagging is a common technique that utilizes sampling to create multiple decision trees from the original dataset, with randomly chosen features used for partitioning at each node. One of the key advantages of Random Forest is its ability to assess feature importance, which has significantly contributed to its widespread application and popularity [24, 25].

The overall feature importance is calculated through the following steps [24]:

1. The importance of node n_j for each node j in a decision tree is computed using the following equation:
2.

$$n_j = W_j C_j - W_{\text{left}(j)} C_{\text{left}(j)} - W_{\text{right}(j)} C_{\text{right}(j)} \quad (2)$$

where W_j represents the probability of reaching node j, and C_j denotes the Gini impurity of that node. The same applies to the left and right child nodes of node j.

3. The importance of each feature (F) within a tree is calculated using the following equation, where m represents the total number of nodes:
4.

$$F(j) = \frac{n_j}{\sum_{i=1}^{m} n_i} \tag{3}$$

5. The overall importance of each feature in a Random Forest (comprising k trees) is determined using the following equation:
6.

$$\text{Feature importance}(i) = \frac{\sum_{j=1}^{m} F(j)}{k} \tag{4}$$

Research [26] suggests that the most important features can vary significantly depending on the chosen method, emphasizing the need to select the appropriate approach for specific analytical goals. In the process of classifying and predicting input data, features within the dataset do not contribute equally to the model's accuracy and performance. Some features play a critical role in the model's final decision, while others may be less relevant or introduce noise, reducing prediction accuracy. To identify the features that most significantly influence model performance, a feature importance analysis was conducted using the built-in importance metric of the Random Forest algorithm, as well as ANOVA and Chi² statistical methods. This analysis was applied across all datasets, with particular emphasis on the highlighted features commonly present in NetFlow records.

Table 7 Overlapping highlighted features selected within the top 50% of features per dataset

Dataset	ANOVA Overlap (Count)	Chi² Overlap (Count)	Random Forest Overlap (Count)
UNSW-NB15	0	0	4
LuFlow	4	5	4
CSE-CIC-IDS2018	3	3	5
CIC-IDS2017	2	2	5

Tab. 7 summarizes the overlap between highlighted features and those selected by ANOVA, Chi², and Random Forest within the top 50% of features for each dataset. Random Forest consistently selected more highlighted features across all datasets (4-5), suggesting its effectiveness in capturing complex or nonlinear feature relationships. In contrast, ANOVA and Chi² performed poorly on the UNSW-NB15 dataset (0 overlaps) but showed better alignment in LuFlow (4-5 overlaps), where statistical relevance was more apparent. For the CSE-CIC-IDS2018 and CIC-IDS2017 datasets, ANOVA and Chi² identified fewer highlighted features (2-3), while Random Forest maintained strong performance. These results indicate that Random Forest is more robust across different data types and may be better suited for feature selection in cybersecurity tasks, especially when feature interactions are complex. Given the consistent and superior overlap of Random Forest with highlighted features across all datasets (Tab. 7), it demonstrates strong capability in identifying relevant features, even when statistical methods like ANOVA and Chi² fall short. This suggests that Random Forest is better at capturing complex, nonlinear relationships typical in network traffic data. Therefore, it is the most suitable choice for a detailed and reliable presentation of feature importance in this analysis.

The following images present graphs that illustrate feature importance in data modelling. These graphs are based on the feature_importances_ parameter, which represents the importance of features assessed using an impurity-based method. This approach utilizes the RF algorithm, implemented in the scikit-learn package [27], which was used for data classification. By analyzing features with this algorithm, it is possible to determine the relative importance of each feature in the context of data prediction and classification, enabling better model interpretation and a deeper understanding of the key factors influencing its accuracy. The y-axis displays feature names, while the x-axis represents their respective importance values. Features highlighted in red correspond to those selected in the dataset variants v2 and v3 and are important for classification. It is worth noting that most of these features are included within the top 50% of all features used in the classification in the original dataset (v1). Their importance varies; however, the majority are positioned in the upper section of the graph - above the threshold marking 50% of the total number of features in the original dataset (v1). This indicates that these features have a greater impact on the model's accuracy and overall performance. The 50% division line represents a threshold that separates features based on their significance to the model. Features above this line are considered more influential for the model's precision and efficiency, while those below the line have a lesser impact. This 50% division line serves as a valuable tool for directing focus toward the most influential features that contribute the most to the decision making process within the model. By utilizing this threshold, it is possible to gain a deeper understanding of which variables exert a dominant influence on modelling outcomes, thereby enabling further enhancement of the model's predictive capabilities and optimization of its performance. The bar chart visualizes feature importance as calculated by a Random Forest model. Features are ranked by their importance, with the most influential at the top. The red bars highlight features that were previously identified as important (i.e., highlighted features), and specifically represent selected features from the full set that match the NetFlow structure, while blue bars represent all other features.

From the chart in Fig. 1, several highlighted features (e.g., dbytes, sbytes, dur, Dpkts) appear prominently above the dashed threshold line, confirming their significant contribution to the model's decisions. This supports the earlier conclusion that Random Forest effectively identifies and prioritizes meaningful features in the dataset.

The bar chart in Fig. 2 illustrates feature importance from a Random Forest model, with red bars indicating highlighted (domain relevant) features. Notably, features such as dest_port, bytes_out, src_port, and duration are

both highlighted and ranked highly, showing strong relevance. In contrast, features like total_entropy and avg_ipt, while important, are not highlighted. The plot confirms that Random Forest effectively identifies key features, aligning well with expert knowledge in several cases. In chart in Fig. 3. several red highlighted features appear among the most influential, such as "Destination Port" and "Bwd Packet Length Std", indicating their strong relevance in the classification task. Most of these red bars are positioned above the dashed threshold line, suggesting that the NetFlow compatible features not only align with domain knowledge but also carry significant discriminative power within the dataset. This underscores their practical value for efficient and interpretable intrusion detection.

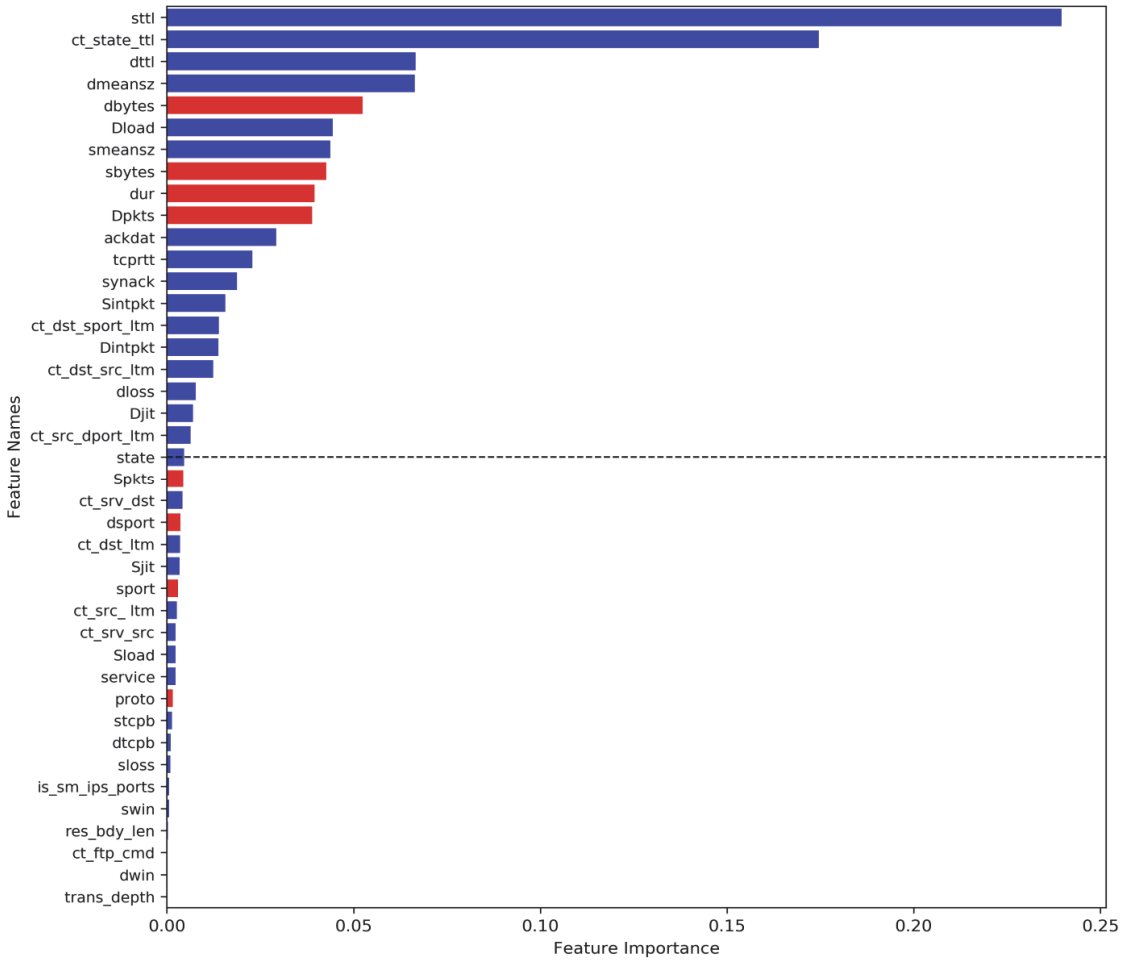

Figure 1 Feature importance in UNSW-NB15 dataset

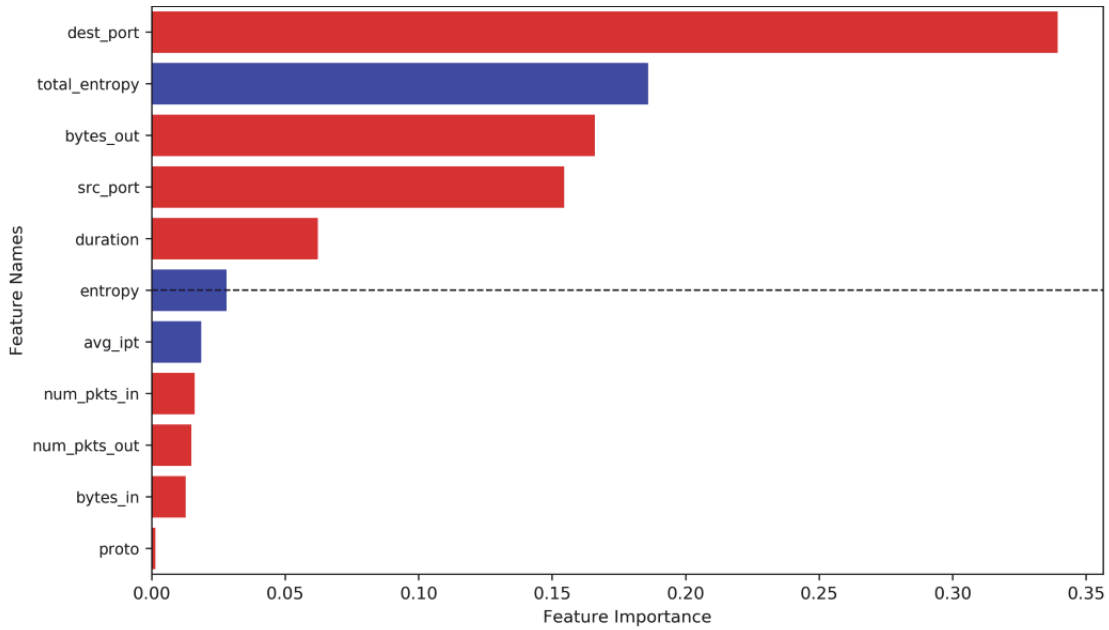

Figure 2 Feature importance in LuFlow dataset

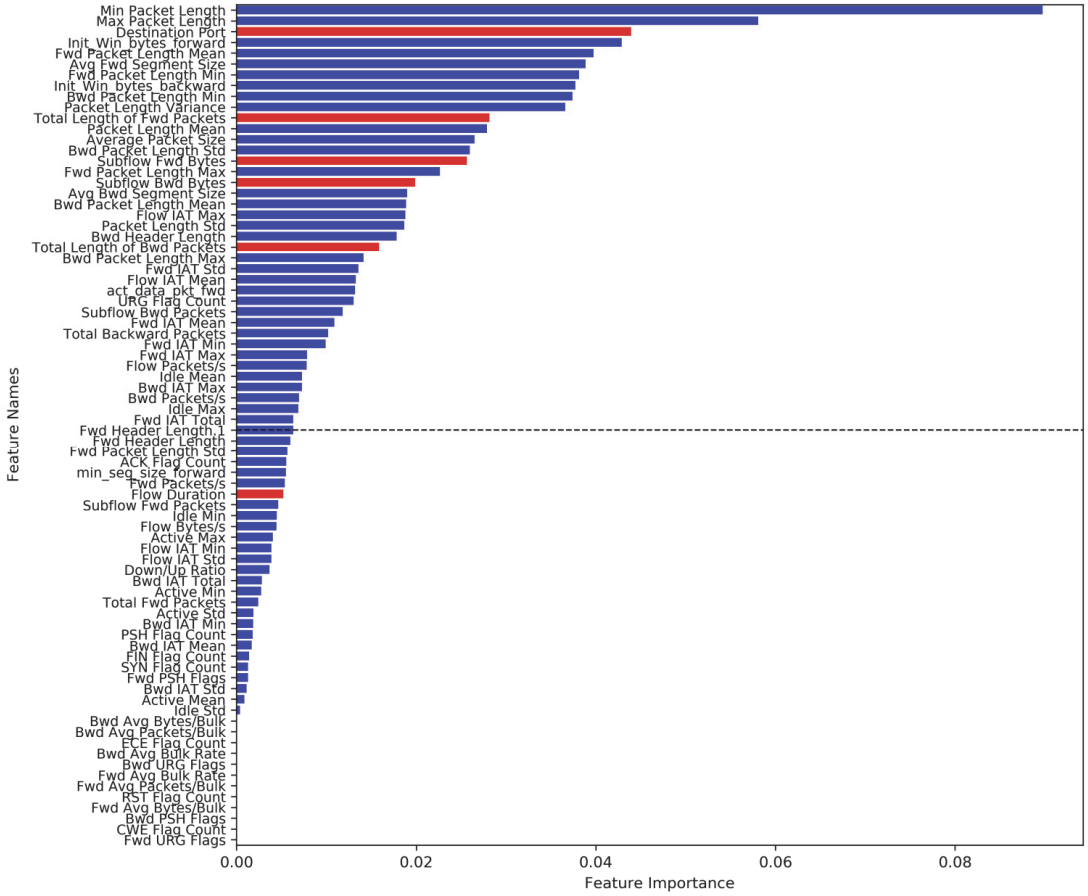

Figure 3 Feature importance in CIC-IDS2017 dataset

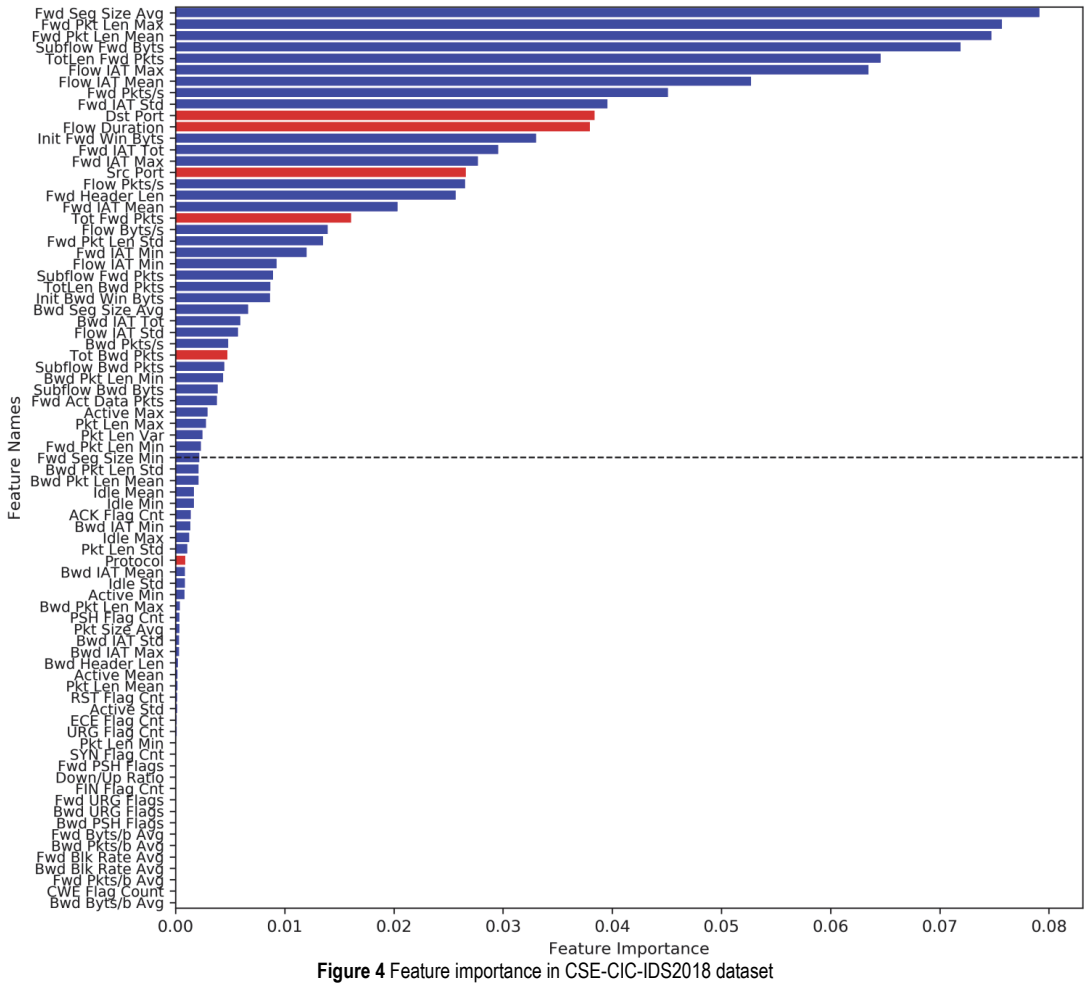

Figure 4 Feature importance in CSE-CIC-IDS2018 dataset

Fig. 4 displays a chart where multiple key features are distinctly emphasized due to their high importance in the classification process. A majority of these features surpass the dashed threshold.

The graphs confirm the validity of feature selection, particularly the alignment between selected features and NetFlow based attributes, during the development of a network traffic classification model. The prominence of NetFlow related features indicates that they play a big role in distinguishing different traffic patterns, reinforcing their relevance in feature engineering for network security and anomaly detection models. Furthermore, the results highlight the importance of feature selection in enhancing model efficiency. The high ranking NetFlow attributes suggest that these features contain significant discriminative information, contributing to improved classification performance.

In Tab. 8, the positions of the 'row_entropy' feature are summarized based on its ranking in feature importance across all datasets adapted to the NetFlow structure (v2).

Table 8 Entropy feature importance summary table

Dataset	Feature Importance Rank of row_entropy	Total Features Ranked	Features (in order of importance)
UNSW-NB15	5th	9	sbytes, Dpkts, dbytes, dur, row_entropy, Spkts, dsport, sport, proto
LuFlow	5th	9	dest_port, bytes_out, src_port, duration, row_entropy, bytes_in, num_pkts_out, num_pkts_in, proto
CSE-CIC-IDS2018	2nd	9	FlowDuration, row_entropy, Dst Port, TotFwdPkts, Protocol, TotBwdPkts, Src Port, BwdByts/b Avg, FwdByts/b Avg
CIC-IDS2017	1st	7	row_entropy, Destination Port, Total Length of FwdPackets, Total Length of BwdPackets, SubflowFwdBytes, SubflowBwdBytes, FlowDuration

The row_entropy feature consistently appears among the top five most important features across all four evaluated datasets, achieving the highest or second highest rank in CIC2017 and CIC2018. This repeated prominence underscores its strong discriminative capability in detecting anomalous traffic behaviour. Although the calculation of 'row_entropy' can be relatively time consuming, its substantial contribution to classification performance justifies its inclusion in the entropy augmented feature sets. As a result, entropy proves to be a valuable and broadly applicable feature, especially when aligning diverse datasets to the NetFlow structure.

4 DISCUSSION

The selection of appropriate datasets is important for developing effective machine learning models for cybersecurity applications. The four datasets presented in this study offer valuable diversity in terms of features, size, and record counts, which can influence model performance. The variety in the number of features and the size of the datasets suggests that a careful balance must be struck between computational complexity and the richness of the data. The datasets provide a solid foundation for further model development, evaluation, and optimization, ensuring their relevance for realworld network security challenges.

Data preparation is often time consuming step in machine learning and data mining projects, especially when working with realworld network traffic data. Proper data cleaning, handling of NaN values, and feature engineering are essential to ensure that datasets are properly formatted and free from noise that could skew model results. The process of balancing class distributions, particularly in the case of network anomaly detection, is vital to improve the accuracy and reliability of machine learning models. Additionally, reducing the number of features to a core set, such as NetFlow features, enhances the comparability and efficiency of classification models. By simplifying the datasets and applying Shannon's entropy as an additional feature, this study ensures better anomaly detection sensitivity while minimizing unnecessary complexity. These data processing and transformation steps are indispensable for achieving reliable and comparable results in the evaluation of machine learning models across different network traffic datasets.

In this study, machine learning techniques were applied to perform classification and anomaly detection across four different datasets, with two variations for each one with reduced features and another with an added entropy feature. The datasets were split into training and testing sets using a 70/30 ratio. The Random Forest algorithm was used for classification, leveraging multiple decision trees to improve prediction accuracy through majority voting. The model's performance was evaluated using key metrics including F2 score, ROC-AUC, Recall, and confusion matrix, with a focus on minimizing False Negative cases. Recall, F2 score, and ROC-AUC collectively provided a robust assessment of the classifier's ability to detect positive cases while considering the trade-off between Recall and Precision. The Pythonbased script efficiently handled all steps from data preprocessing and model training to performance evaluation, ensuring an indepth analysis of the classification effectiveness and anomaly detection capability.

With a comparison of four datasets (UNSW-NB15, LuFlow, CIC-IDS2017, and CSE-CIC-IDS2018) across different feature sets (Original (v1), NetFlow (v2), and Entropy (v3)), with key classification metrics such as Accuracy Score, True Negatives (TN), False Positives (FP), False Negatives (FN), True Positives (TP), Recall, F2 score, and Precision-Recall AUC. The Entropy (v3) versions demonstrate competitive or even better FN values compared to the NetFlow (v2) models. For example, in the UNSW-NB15 dataset, the Entropy (v3) model achieves an FN of 317, while NetFlow (v2) has an FN of 359, showing an improvement. In the LuFlow dataset, the Entropy (v3) model has an FN of 16, compared to 28 for NetFlow (v2). In the CIC-IDS2017 dataset, the Entropy (v3) model shows an FN of 333, slightly better than the 355 for NetFlow (v2). Lastly, in the CSE-CIC-IDS2018 dataset, the Entropy (v3)

model achieves an FN of 70, which is worse than the Original (v1) version's FN of 8, while NetFlow (v2) has an FN of 56. While the Entropy (v3) models show slightly worse or comparable results in other metrics like accuracy, Recall, F2 score, and Precision-Recall AUC, the differences are minimal. The Entropy (v3) models, with an PR-AUC accuracy metric, demonstrate that they can maintain high classification performance with a reduced feature set. This makes them a strong choice when feature reduction is important without significantly compromising the model's effectiveness. The correspondence between the selected features and NetFlow attributes suggests that incorporating domain specific knowledge in feature selection can significantly enhance classification accuracy. The presence of NetFlow related features above the 50% importance threshold indicates that these attributes effectively capture key network traffic patterns, making them essential for network traffic classification models.

5 CONCLUSION

This study highlights the importance of dataset diversity, as the four datasets analyzed here vary in features, size, and record counts, all of which influence model performance. The data preparation process, including cleaning, handling missing values, and balancing class distributions, proves essential for ensuring the accuracy and reliability of the models. Reducing the feature set and adding Shannon's entropy feature enhances the model's sensitivity to anomalies while keeping the complexity manageable. The main contributions of this paper can be summarized in several key points. Firstly, datasets used for testing and anomaly detection are often enriched with many irrelevant features, and the methods used to derive these features are frequently undocumented. One of the analyzed datasets (LUFlow) includes an entropy-based feature, which plays a significant role in classification and anomaly detection. However, the procedure for reproducing this feature in other datasets - particularly in realworld datasets collected directly from network devices, such as NetFlow - is not adequately described. This paper provides a method for calculating Shannon entropy within NetFlow adapted datasets across all four evaluated datasets. Extensive testing was conducted using eight different machine learning algorithms, and based on the obtained results, the most suitable algorithm for classification was identified.

When applying machine learning techniques, particularly the Random Forest algorithm, the results reveal that Entropy (v3) models consistently show competitive or improved performance, especially in reducing False Negative (FN) values. Ultimately, the findings of this study demonstrate that Entropy (v3) models are highly effective for anomaly detection, offering excellent performance with a reduced feature set. This makes them a valuable choice in scenarios where feature reduction is crucial, without sacrificing model efficacy. The results indicate that feature selection and dataset optimization are key to achieving the best balance between model simplicity and detection accuracy.

6 REFERENCES

[1] Yang, L., Cai, M., Duan, Y., & Yang, X. (2019). Intrusion detection based on approximate information entropy for random forest classification. *Proceedings of the 2019 4th International Conference on Big Data and Computing (ICBDC 2019)*, 125-129. https://doi.org/10.1145/3335484.3335488

[2] Timcenko, V. & Gajin, S. (2021). Machine learning enhanced entropy-based network anomaly detection. *Advances in Electrical and Computer Engineering*, *21*(4), 51-60. https://doi.org/10.4316/AECE.2021.04006

[3] Niknami, N. & Wu, J. (2022). Entropy-KL-ML: Enhancing the entropy-KL-based anomaly detection on software-defined networks. *IEEE Transactions on Network Science and Engineering*, *9*(6), 4458-4467. https://doi.org/10.1109/TNSE.2022.3202147

[4] Mohammad, R., Saeed, F., Almazroi, A. A., Alsubaei, F. S., & Almazroi, A. A. (2024). Enhancing intrusion detection systems using a deep learning and data augmentation approach. *Systems*, *12*(3), 79. https://doi.org/10.3390/systems12030079

[5] Zaman, M., Upadhyay, D., & Lung, C.-H. (2023). Validation of a machine learning-based IDS design framework using ORNL datasets for power system with SCADA. *IEEE Access*, *11*, 118414-118426. https://doi.org/10.1109/ACCESS.2023.3326751

[6] Shin, S., Lee, I., & Choi, C. (2019). Anomaly dataset augmentation using the sequence generative models. *Proceedings of the 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1143-1148. https://doi.org/10.1109/ICMLA.2019.00190

[7] Elejla, O. E., Anbar, M., Hamouda, S., Belaton, B., Al-Amiedy, T. A., & Hasbullah, I. H. (2022). Flow-based IDS features enrichment for ICMPv6-DDoS attacks detection. *Symmetry*, *14*(12), 2556. https://doi.org/10.3390/sym14122556

[8] Quincozes, S. E., Albuquerque, C., Passos, D., & Mossé, D. (2024). ERENO: A framework for generating realistic IEC-61850 intrusion detection datasets for smart grids. *IEEE Transactions on Dependable and Secure Computing*, *21*(4), 3851-3865. https://doi.org/10.1109/TDSC.2023.3336857

[9] Janarthanan, T. (2017). Feature selection in UNSW-NB15 and KDDCUP '99 datasets.

[10] Mills, R. (2023). LUFlow network intrusion detection data set. https://www.kaggle.com/datasets/mryanm/luflow-network-intrusion-detection-data-set/data

[11] LUFlow network intrusion detection data set. (2022). https://www.kaggle.com/datasets/mryanm/luflow-network-intrusion-detection-data-set

[12] Chua, T.-H., & Salam, I. (2022). Evaluation of machine learning algorithms in network-based intrusion detection system. http://arxiv.org/abs/2203.05232

[13] CSE-CIC-IDS2018 on AWS. (2022). https://www.unb.ca/cic/datasets/ids-2018.html

[14] Canadian Institute for Cybersecurity. (2025). Intrusion detection evaluation dataset (CIC-IDS2017). https://www.unb.ca/cic/datasets/ids-2017.html

[15] Khan, Z. I., Afzal, M. M., & Shamsi, K. N. (2024). A comprehensive study on CIC-IDS2017 dataset for intrusion detection systems. *International Research Journal on Advanced Engineering Hub, 2*(2), 254-260. https://doi.org/10.47392/IRJAEH.2024.0041

[16] Rosay, A., Cheval, E., Carlier, F., & Leroux, P. (2022). Network intrusion detection: A comprehensive analysis of CIC-IDS2017. *Proceedings of the 8th International Conference on Information Systems Security and Privacy*, 25-36. https://doi.org/10.5220/0010774000003120

[17] Sehrawat, R. Data preparation. https://www.academia.edu/attachments/62900205/download_file
[18] Data science report. (2022). https://visit.figure-eight.com/2015-data-scientist-report.html
[19] Haghighat, M. H., & Li, J. (2018). Edmund. *Proceedings of the 8th International Conference on Communication and Network Security*, 1-6. https://doi.org/10.1145/3290480.3290484
[20] Machine learning classification algorithm. https://www.javatpoint.com/machine-learning
[21] Hasan, M. A. M., Nasser, M., Pal, B., & Ahmad, S. (2014). Support vector machine and random forest modeling for intrusion detection system (IDS). *Journal of Intelligent Learning Systems and Applications*, *6*(1), 45-52. https://doi.org/10.4236/jilsa.2014.61005
[22] Stąpor, K. (2018). Evaluating and comparing classifiers: Review, some recommendations and limitations. https://doi.org/10.1007/978-3-319-59162-9_2
[23] Olamendy, J. C. (2025). Unveiling the power of metrics in classification: A comprehensive guide.
[24] Al Sagri, H. & Ykhlef, M. (2020). Quantifying feature importance for detecting depression using random forest. *International Journal of Advanced Computer Science and Applications*, *11*(5). https://doi.org/10.14569/IJACSA.2020.0110577
[25] Gebreyesus, Y., Dalton, D., Nixon, S., De Chiara, D., & Chinnici, M. (2023). Machine learning for data center optimizations: Feature selection using Shapley additive explanation (SHAP). *Future Internet*, *15*(3), 88. https://doi.org/10.3390/fi15030088
[26] Saarela, M. & Jauhiainen, S. (2021). Comparison of feature importance measures as explanations for classification models. *SN Applied Sciences*, *3*(2), 272. https://doi.org/10.1007/s42452-021-04148-9
[27] Random Forest Classifier (2025). https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Contact information:

Igor FOSIĆ, PhD
(Corresponding author)
HEP-Telekomunikacije d.o.o.,
M. Divalta 199, HR-31000 Osijek
E-mail: igor.fosic@hep.hr

Drago ŽAGAR, PhD, Full Professor
Josip Juraj Strossmayer University of Osijek,
Faculty of Electrical Engineering, Computer Science and Information Technology Osijek,
Kneza Trpimira 2B, HR-31000 Osijek
E-mail: drago.zagar@ferit.hr