INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 1, PP. 1–8
Manuscript received August 4, 2025; revised February, 2026. doi: 10.24425/ijet.2026.157880

Data exploration algorithms in anomaly detection in communication protocols

Michał Kaczmarczyk, Kacper Kocemba, Maciej Stranz, Mateusz Winnicki, and Sebastian Plamowski

***Abstract*—Cybersecurity in modern communication networks is of paramount importance, particularly in critical infrastructure sectors. Anomaly detection in communication protocols is a key component in identifying and mitigating cyber threats. This study explores data-centric approaches for anomaly detection using machine learning algorithms. We evaluate the effectiveness of ensemble models incorporating Isolation Forest, XGBoost, and Autoencoders to reduce false positives while maintaining high accuracy. Our methodology involves training on both labeled and unlabeled datasets, including NSL-KDD and CIC-IDS2017, to simulate real-world attack scenarios. Experimental results demonstrate that the proposed ensemble learning approach enhances detection performance, offering a balanced trade-off between precision and false alarm reduction. These findings contribute to the development of robust and scalable intrusion detection systems suitable for deployment in industrial and critical infrastructure networks.**

***Keywords*—Anomaly detection; communication protocols; cybersecurity; machine learning; intrusion detection; ensemble learning**

I. Introduction

CYBERSECURITY is a critical aspect of modern information systems, particularly in the context of critical infrastructure. The ability to quickly and accurately detect attacks is essential for network security as contemporary threats become increasingly complex. This need is especially pressing in Industrial Control Systems (ICS), which form the backbone of critical infrastructure in the energy, manufacturing, and transportation sectors [1].

The history of the first cyberattack dates back to 1834 when the Blanc brothers (with the help of bribed telegraph operators) manipulated the operation of the world's first optical telegraph [2] launched in France. Their idea was to insert hidden information about directions in the financial market into the transmitted government information, which allowed them to access information much faster. The mechanism used was based on the premise of steganography [3], where vital information was hidden unnoticed in the transmitted content. This attack shows that hacking, manipulation, and deception have accompanied humans for centuries, and there is no indication that they will stop.

Authors are with Warsaw University of Technology, Faculty of Electronics and Information Technology, Institute of Control and Computational Engineering, Poland (e-mail: {michal.kaczmarczyk2.stud, kacper.kocemba.stud, maciej.stranz.stud, mateusz.winnicki2.stud, sebastian.plamowski}@pw.edu.pl).

The modern history of cyber attacks begins in the mid-1980s of the previous century. The first known attacks were the Cascade virus, the Morris worm, or the Melissa virus, released a decade [4]. The initial attacks were characterized by randomness, with attacks targeting personal computers. With the proliferation of mobile devices, the boundaries of accessibility and security have been significantly pushed. Attacks have become much more sophisticated, consisting of multi-stage consecutive attack phases aimed at gaining full access to specific resources on the network (kill chain attack [5]).

In addition to attacks in the IT area, the second decade of the 21st century has seen many attacks on computer networks (IT and OT networks) of enterprises classified as critical infrastructure, such as power plants, transmission networks (water, gas, power) mines, petrochemicals and other [6]. Some of the best-known include Stuxnet (2010). [7], BlackEnergy (2015) [8], WannaCry (2017) [9], NotPetya (2017) [10], Colonia Pipeline (2021) [11], SolarWinds (2020) [12], Hydro Attack (2019) [13], Oldsmar Water Treatment Plant (2021) [14]

Cyber attacks on critical infrastructure are becoming increasingly sophisticated and pose a real threat to public security, the economy and the functioning of states. Their scale and impact underscore the need for investment in advanced protection technologies, international cooperation, and threat awareness-building. The growing number of successful attacks against ICS underscores the urgent need to create defense mechanisms capable of accurately and timely detecting anomalies [15].

Among security mechanisms are top-level mechanisms such as protecting email, isolating and analyzing suspicious files, encrypting sensitive data, blocking access to suspicious sites, detecting and blocking exploits and malware, regularly updating software, limiting user and device permissions to a minimum level, and allowing only approved applications to run [16]–[18]. The purpose of these mechanisms is to prevent attacks.

Unfortunately, these mechanisms do not always provide full protection, and the continuous improvement of attack techniques allows attackers to break through them effectively. As a result, attackers get inside the organization and move on to the next stages of attacks. The key is to detect this phase as soon as possible, even before the damage is done. To this end, an effective way is to detect cybercriminals based on observation of network traffic parameters using various

methods to analyze network traffic to detect suspicious activity [19].

There are many analysis techniques, also based on machine learning [20]. They are mainly based on deep package inspection (DPI) [21] or the aforementioned analysis of traffic parameters. The monitored data are compared with the learned models; on this basis, anomalies that may be a symptom of an attack are detected.

Data-centric approaches are not without their drawbacks. The main problem is the lack of data from attacks. In real-world installations, a great deal of normal traffic data is available, and usually, there is no data from attacks, or if there is, it is very little, covering the entire range of possible techniques used by cyber criminals. The second problem is false alarms, which can lead to unnecessary interventions, wasting time and resources [22]. Given the context of communication protocols, even a small percentage of false alarms can result in a significant number of misleading alarms. Therefore, in this article, we focus our research on comparing results obtained on labeled and unlabeled samples and pay special attention to false alarms.

Given the context of communication protocols, even a small percentage of false alarms can result in a significant volume of misleading alarms. Therefore, in this paper, we focus our research on labeled and unlabeled samples when we assume that we only know the data without attacks, and on such learned models, we try to detect anomalies in the data.

II. Related work

A review of the existing literature shows that many studies have focused on analyzing popular datasets we will use in our study. Researchers used various machine learning methods. For example, in [23] authors achieved approximately 96% accuracy using Random Forest and neural networks. Similarly, in [24], authors used PCA for preprocessing and classified attacks with DNN, LSTM, and CNN, reporting respective accuracies of 94. 61%, 97.67% and 98.61%. Other works, such as [25], used SMOTE and PCA to improve AdaBoost performance, producing precision and recall of 81.83% and 100%, respectively. Finally, [26] demonstrated that Recurrent Neural Networks (RNNs) excel at intrusion detection, outperforming CNNs, and Naive Bayes by maintaining a better balance between precision and recall.

In the context of the NSL-KDD dataset, researchers have progressively tried to achieve improved performance in intrusion detection employing machine learning methods. For instance, the 2015 study reported detection rates of 81.2% for intrusion detection and 79.9% for attack type classification tasks [27]. Six years later, in authors [28] demonstrated significant advancements by employing recursive feature elimination to select optimal features and using deep neural networks (DNN) and recurrent neural networks (RNN) for classification. This approach achieved a detection accuracy of 94%. DNN was applied for binary classification (normal vs. attack), while RNN was utilized for multi-class classification (Normal, DoS, Probe, R2L, U2R).

In [29] the authors point out the problem of high false positive rates when using anomaly-based models. Anomaly-based models classify an unobservable pattern as a hazard, where it may be normal, but is not included in the training dataset. The result is an over-fitting model lacking the generalization feature. The authors suggest using a deep model instead of traditional models because it can be generalized more easily. Big data and deep models resulted in a lower percentage of false-positive trials. Experiments conducted on the NSL-KDD benchmark using deep learning instead of traditional learning show a 10% lower percentage of false positives,

The authors used a different approach in [30], their method is to test for significance using eXplainable Artificial Intelligence (XAI) and, along with a confidence measure, identify detections that are more likely to be false. The authors showed that using the LYCOS-IDS2017 dataset, it is possible to eliminate more than 65% of all false alarms, with a loss of only 0.38% of true alarms. Conversely, using only the confidence measure, the elimination of false positives is about 50%, with a loss of 0.42% of true positives.

Although the studies reviewed provided valuable information and showed high accuracy rates, most prioritized detection accuracy, the topic of false alarms is far less frequently addressed. It is an ongoing problem in real systems. High false positive rates can overwhelm security analysts, potentially delaying responses to genuine threats. In addition, the reviewed studies predominantly relied on individual models rather than integrating multiple approaches, leaving room for improvement in the development of robust and versatile intrusion detection systems. In our work we use predefined training and test sets to ensure a fair and realistic evaluation of the performance of our models.

A. Article Contribution

Previous studies have demonstrated the potential of machine learning methods in intrusion detection using the CIC-IDS2017 and NSL-KDD datasets; however, they mainly focused on achieving high detection accuracy without adequately addressing critical limitations, such as managing false positive rates and exploring approaches more complex than evaluating different machine learning methods.

Our research seeks to address these gaps by proposing an ensemble learning approach that combines the strengths of Isolation Forest, XGBoost, and Support Vector Machines (SVM). This method aims to reduce false positives while maintaining or exceeding the high accuracy benchmarks set by previous studies. By integrating diverse machine learning techniques, our approach not only enhances precision and recall, but also ensures a more balanced and reliable intrusion detection system capable of real-world deployment.

We will evaluate our ensemble model in two scenarios: the first involves training on labeled data with both normal flow and attack samples, while the second uses only normal flow data without labels. This second case reflects real-world situations where labeled attack data may be unavailable, challenging the model to identify anomalies effectively.

III. Methodology

A. Datasets analisys

In this paper, we present the results of our experiments, which evaluate the effectiveness of this approach using two popular datasets: NSL-KDD[1] and CIC-IDS2017[2]. These data sets provide diverse representations of network traffic, including both normal and anomalous activity, enabling a comprehensive evaluation of the proposed method.

1) NSL-KDD dataset: The NSL-KDD dataset is a widely used benchmark dataset for intrusion detection research. It was created as an improved version of the original KDD Cup 1999 dataset, which has been one of the most commonly used datasets in the field of cybersecurity. It consists of two datasets: KDDTrain+ and KDDTest+. We used KDDTrain+ to train our models and evaluated them on KDDTest+. Both datasets together consist of 148517 rows and 43 features.

2) CIC-IDS2017 dataset: The CIC-IDS2017 dataset is a comprehensive dataset designed for research in intrusion detection and cybersecurity. It was created by the Canadian Institute for Cybersecurity (CIC) and is widely used to evaluate the performance of intrusion detection systems (IDS) in detecting modern network attacks. The entire CIC-IDS2017 dataset consists of eight smaller subsets, each containing data collected over different days and at various times. In our research, we combined all subsets, and after preprocessing and feature engineering we split it into train, and test datasets in ratio 70:30.

B. Data preprocessing

In the process of preparing the dataset for network traffic analysis, we undertook several critical data cleaning and preprocessing steps. These measures ensured the integrity, consistency, and suitability of the data for subsequent machine learning and statistical analyses. Below, we detail the specific actions carried out during this phase.

1) Duplicates removal: We identified and removed duplicate records from the dataset to enhance its integrity and reduce redundancy. Duplicate rows, if left unaddressed, could have biased the analysis by overrepresenting specific patterns in the data. This could lead to inflated importance of repeated observations and inaccurate modeling outcomes. By eliminating duplicates, we ensured that the dataset more accurately represented the underlying patterns of network traffic, providing a clean and streamlined foundation for analysis.

2) Handling missing values: To address missing values, we carefully examined the dataset and identified key features, which contained gaps. Missing data, if not handled, can disrupt machine learning algorithms and distort statistical analyses. We replaced these missing values with the median of their respective columns. Alternatively, we also conducted tests by removing the missing data, and the results were almost identical.

[1] https://www.kaggle.com/datasets/hassan06/nslkdd
[2] https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset

3) Handling infinite values: During the inspection of numerical features, we encountered infinite and undefined values, which could have disrupted calculations and caused instability in machine learning models. These values were replaced with placeholders to signify missing data. Subsequently, appropriate measures, such as imputation with the median, were applied to restore meaningful values. This ensured that all numerical features were numerically stable and ready for further processing. This step was crucial in creating a dataset that was both reliable and computationally manageable.

4) Balance of classes: The same preprocessing techniques were used for both datasets. To adapt the them for binary classification, we re-labeled all entries originally categorized as attacks into a single „attack" class, while the 'normal' labels were left unchanged. The next step involved evaluating the balance of classes within the datasets to ensure an appropriate distribution for binary classification. The results have been shown on the images 1 and 2. Due to the poorer class balance in CIC-IDS2017, training in this data set presents a greater challenge. However, NSL-KDD is much better balanced. Subsequently, we used one-hot encoding for categorical features and scaled numeric features. Additionally we changed text labels to 1 for „normal" class and 0 for „attack" class.

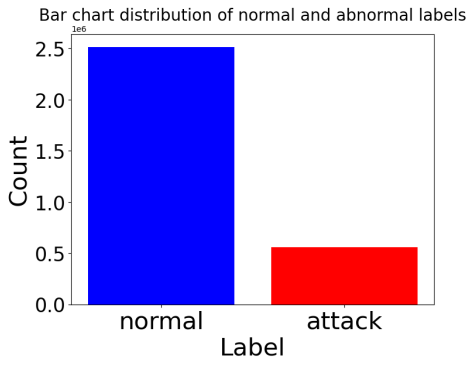

Fig. 1. Class distribution in CIC-IDS2017 dataset

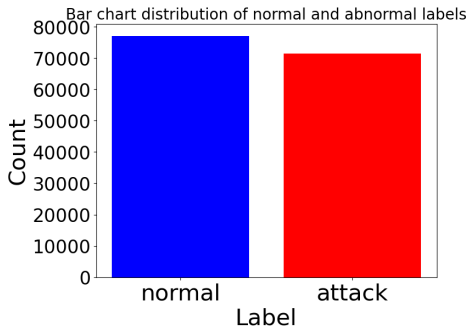

Fig. 2. Class distribution in NSL-KDD dataset

5) Correlation: Analyzing feature correlations was a key step in understanding the relationships between variables and selecting the most relevant predictors for attack detection. We constructed a correlation matrix to measure the strength and direction of linear relationships between numerical features.

Features with strong positive or negative correlations with the target variable were highlighted as potential predictors. These features are likely to have a significant impact on the detection and classification of attacks.

C. Feature engineering

For both datasets, we used Principal Component Analysis (PCA) to select the best features for model training. Dimensionality reduction was a key step in preprocessing the dataset, aimed at improving computational efficiency and enhancing model performance. With a high-dimensional dataset, the risk of overfitting and computational challenges increases significantly. To address this, we employed Incremental Principal Component Analysis (IncrementalPCA) to reduce the dimensionality of the feature set while retaining as much relevant information as possible. PCA is a linear algebra-based method aimed at transforming the original features into a new coordinate system, referred to as principal components. These principal components are linear combinations of the input variables, capturing the maximum variance in the data. The variance of each subsequent component decreases, meaning that the first principal component contains the most information about the data's variability.

Advantages:

- **Memory efficiency:** By processing data in batches, IPCA requires minimal RAM usage, as only the current data batch and the result matrix are stored at any given time.
- **Suitability for Large Datasets:** IPCA enables the analysis of datasets too large to be loaded into memory all at once.
- **Scalability:** The algorithm can easily adapt to streaming data analysis, where data arrives continuously.

Disadvantages:

- **Approximate results:** IPCA results may be less accurate than those of classical PCA, as the covariance matrix is updated iteratively rather than computed for the entire dataset simultaneously.
- **Dependency on batch size:** Choosing a batch size that is too small may increase computation time, while a batch size that is too large can lead to memory issues.
- **Limited parameter modification:** During the algorithm's execution, it is not easy to adjust the number of principal components or the batch size. We used grid search to find the parameters of the best models for the validation set. The F1-score was employed as the primary selection criterion, as it provides a balanced measure of precision and recall.

D. Applied machine learning methods

The core of our approach lies in ensemble learning, wherein we integrate predictions from multiple models. Specifically, we employ Isolation Forest, XGBoost, and Support Vector Machines (SVM) due to their complementary strengths in detecting anomalies and classifying network activity. The ensemble decision process is based on a majority vote: an instance is labeled as anomalous only if at least two out of the three models classify it as such. In addition to using Isolation Forest and XGBoost, our ensemble learning approach also incorporates Autoencoders for anomaly detection. Autoencoders are a type of neural network architecture designed for unsupervised learning. They consist of an encoder and a decoder: the encoder compresses the input data into a lower-dimensional representation, while the decoder reconstructs the data back to its original form. The network is trained to minimize the reconstruction error, and data points with high reconstruction error are flagged as anomalies. By utilizing Autoencoders, we can capture complex, non-linear patterns in the data that other models might miss. Isolation Forest, on the other hand, is a tree-based anomaly detection algorithm. It isolates anomalies by recursively partitioning the data using randomly selected features. Since anomalies are fewer and different from normal data points, they are easier to isolate in fewer splits. The algorithm assigns an anomaly score based on the number of splits required to isolate a data point.

The combination of these models — Isolation Forest for tree-based anomaly detection, Autoencoders for capturing non-linear patterns, and XGBoost for classification — providdodaes a robust ensemble that leverages the strengths of each model. The ensemble decision process, as mentioned earlier, is based on a majority vote, ensuring that an instance is classified as anomalous only if at least two out of the three models classify it as such. This reduces false positives and enhances the overall detection accuracy

E. Hyperparameters calibration

Hyperparameter optimization for each model was performed using grid search and cross-validation on the training sets of both datasets. Model-specific configuration details, including contamination rates for Isolation Forest and size of Autoencoder, will be reported in subsequent sections. Hyperparameter optimization for Isolation Forest and Autoencoder was performed using grid search and cross-validation on the training sets.

Isolation Forest: The optimized key parameters include the contamination rate (proportion of anomalies in the dataset) and the number of estimators (trees in the forest). The optimal configuration ensures a balance between detection sensitivity and false positives.

Autoencoder: The architecture was fine-tuned by varying the number of layers, neurons per layer, and activation functions. Furthermore, the learning rate and batch size were optimized to achieve minimal reconstruction error during training.

F. Full research procedure

Our experimental procedure involves the following steps:

- Training individual models on the preprocessed NSL-KDD and CIC-IDS2017 training sets.
- Evaluating individual model performance on the respective test sets to establish baselines.
- Combining predictions using the ensemble voting mechanism and evaluating integrated performance. majority voting refereed to as hard voting was employed to aggregate the predictions of individual classifiers.

- Analyzing results based on metrics such as accuracy, precision, recall, F1-score, and false positive rate.

Moreover, we also conducted a more realistic study, which means that we trained the models exclusively on data originating from normal traffic. In this scenario, we selected only data labeled as „normal" for training set, and evaluated models on the original test set with both „normal" and „attack" instances. Apart from this modification, in this section, we followed the procedures outlined above.

IV. Results

This section presents the results obtained from training different anomaly detection models on two datasets: NSL-KDD and CIC-IDS2017. The evaluated models include Isolation Forest (IF), Autoencoder (AE), a hybrid model combining IF and AE, XGBoost, and a Voting Classifier. We report the confusion matrices and key performance metrics for each model.

A. Training on full datasets

In this scenario, models were trained on the entire dataset, containing both normal and attack traffic. This setup allows models to learn patterns from both classes and assess their performance on detecting intrusions.

1) NSL-KDD:

Best Model-Specific Configuration Details

- **Isolation Forest:**
  - Contamination Rate: 5.0%
  - Autoencoder Size: N/A
  - Filters: N/A
  - Other Parameters: n_estimators: 100
- **Autoencoder:**
  - Contamination Rate: N/A
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: N/A
- **Hybrid Model (IF + AE):**
  - Contamination Rate: 10.0%
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: n_estimators: 100
- **XGBoost:**
  - Contamination Rate: N/A
  - Autoencoder Size: N/A
  - Filters: Max Depth: 10, Learning Rate: 0.1
  - Other Parameters: n_estimators: 100, eta: 0.05, objective: reg:tweedie

TABLE I
Performance comparison of anomaly detection models trained on NSL-KDD dataset based on confusion matrix, accuracy, and false positives

Model	Confusion Matrix	Accuracy	FP
XGBoost	$\begin{bmatrix} 11784 & 1049 \\ 3005 & 6706 \end{bmatrix}$	0.82	3005
Isolation Forest	$\begin{bmatrix} 9003 & 3830 \\ 816 & 8895 \end{bmatrix}$	0.79	816
Autoencoder	$\begin{bmatrix} 8820 & 4013 \\ 436 & 9275 \end{bmatrix}$	0.80	436
Voting Classifier	$\begin{bmatrix} 4102 & 8731 \\ 128 & 9583 \end{bmatrix}$	0.61	128

2) CIC-IDS2017:

Best Model-Specific Configuration Details

- **Isolation Forest:**
  - Contamination Rate: 16.8%
  - Autoencoder Size: N/A
  - Filters: N/A
  - Other Parameters: n_estimators: 200
- **Autoencoder:**
  - Contamination Rate: N/A
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: N/A
- **Hybrid Model (IF + AE):**
  - Contamination Rate: 10.0%
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: n_estimators: 100
- **XGBoost:**
  - Contamination Rate: N/A
  - Autoencoder Size: N/A
  - Filters: Max Depth: 6, Learning Rate: 0.1
  - Other Parameters: n_estimators: 100, eta: 0.05, objective: reg:tweedie

TABLE II
Performance comparison of anomaly detection models trained on full CIC-IDS2017 dataset based on confusion matrix, accuracy, and false positives

Model	Confusion Matrix	Accuracy	FP
XGBoost	$\begin{bmatrix} 126487 & 1276 \\ 39 & 627691 \end{bmatrix}$	0.9979	39
Isolation Forest (trained on ATTACK + BENIGN)	$\begin{bmatrix} 19957 & 107805 \\ 109842 & 519977 \end{bmatrix}$	0.6884	109842
Autoencoder	$\begin{bmatrix} 112038 & 15825 \\ 74961 & 554981 \end{bmatrix}$	0.8796	74961
Voting Classifier	$\begin{bmatrix} 96021 & 31742 \\ 9514 & 619432 \end{bmatrix}$	0.9321	9514

For the **NSL-KDD** dataset, XGBoost achieved the highest overall accuracy at 82%, albeit with a relatively high false positive count (FP = 3005). The Autoencoder model demonstrated a balanced performance, yielding an accuracy of 80% and the lowest false positive rate (FP = 436), indicating strong discriminatory capability. Isolation Forest attained a comparable

accuracy of 79%, with a slightly higher false positive rate (FP = 816). The Voting Classifier, while maintaining a low false positive count (FP = 128), suffered from reduced accuracy (61%), suggesting limited effectiveness in this context.

On the **CIC-IDS2017** dataset, XGBoost significantly outperformed the other models, achieving near-perfect accuracy (99%) and a negligible false positive rate (FP = 39). In contrast, Isolation Forest and Autoencoder models exhibited lower accuracy (69% and 88%, respectively) and high false positive rates (FP = 109842 and FP = 74961, respectively). In contrast to the previous dataset, Voting Classifier showed a performance of 93% and a false positive rate (FP = 9514). This is worse than XGBoost, but better than Isolation Forest and Autoencoder.

Overall, the results indicate that XGBoost consistently provides better performance in both data sets, particularly evident in the second set where it was better at both accuracy and false positive rates.

The use of a voting classifier in the first data set improved the false positive rate at the expense of the accuracy. In the second data set, the voting classifier is clearly worse than XGBoost and better than Isolation Forest and Autoencoder.

The use of the XGBoost classifier is not always possible, as it requires labeled data, which is not always available in real-world conditions. Typically, attack data is not known, so the use of XGBoost is limited. Results for a situation in which normal traffic is available are presented in the next subsection.

B. Training on normal activity

In this experiment, models were trained only on normal traffic data, without exposure to attack patterns during training. This setup tests their ability to detect anomalies in an unsupervised manner.

1) NSL-KDD:

BEST MODEL-SPECIFIC CONFIGURATION DETAILS

- **Isolation Forest:**
  - Contamination Rate: 3.0%
  - Autoencoder Size: N/A
  - Filters: N/A
  - Other Parameters: n_estimators: 100
- **Autoencoder:**
  - Contamination Rate: N/A
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: N/A
- **Hybrid Model (IF + AE):**
  - Contamination Rate: 5.0%
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: n_estimators: 100

TABLE III
PERFORMANCE COMPARISON OF ANOMALY DETECTION MODELS TRAINED ON NORMAL ACTIVITY FROM NSL-KDD DATASET BASED ON CONFUSION MATRIX, ACCURACY, AND FALSE POSITIVES

Model	Confusion Matrix	Accuracy	FP
Isolation Forest	$\begin{bmatrix} 5381 & 7452 \\ 336 & 9375 \end{bmatrix}$	0.66	336
Autoencoder	$\begin{bmatrix} 9029 & 3804 \\ 533 & 9178 \end{bmatrix}$	0.81	533
Voting Classifier	$\begin{bmatrix} 5381 & 7452 \\ 36 & 9675 \end{bmatrix}$	0.67	36

2) CIC-IDS 2017:

BEST MODEL-SPECIFIC CONFIGURATION DETAILS

- **Isolation Forest:**
  - Contamination Rate: 16.8%
  - Autoencoder Size: N/A
  - Filters: N/A
  - Other Parameters: n_estimators: 200
- **Autoencoder:**
  - Contamination Rate: N/A
  - Autoencoder Size: 128-64-32-64-128
  - Filters: N/A
  - Other Parameters: N/A
- **Hybrid Model (IF + AE):**
  - Contamination Rate: 5.0%
  - Autoencoder Size: 1024-512-256-128-64-128-256-512-1024
  - Filters: N/A
  - Other Parameters: n_estimators: 100

TABLE IV
PERFORMANCE COMPARISON OF ANOMALY DETECTION MODELS TRAINED ON NORMAL ACTIVITY FROM CIC-IDS2017 DATASET BASED ON CONFUSION MATRIX, ACCURACY, AND FALSE POSITIVES

Model	Confusion Matrix	Accuracy	FP
Isolation Forest	$\begin{bmatrix} 125266 & 2497 \\ 553795 & 75151 \end{bmatrix}$	0.26	553795
Autoencoder	$\begin{bmatrix} 114986 & 12777 \\ 62895 & 566051 \end{bmatrix}$	0.90	62895
Voting Classifier	$\begin{bmatrix} 98446 & 29317 \\ 5180 & 623766 \end{bmatrix}$	0.95	5180

For the **NSL-KDD dataset**, the Autoencoder achieved the highest accuracy (0.81), albeit with a moderate false positive rate (FP = 533). The Voting Classifier reported the lowest false positive rate (FP = 36), but exhibited slightly lower accuracy (0.67). Isolation Forest showed limited performance with the lowest accuracy (0.66), though its false positive count (FP = 336) remained moderate.

In the case of the **CIC-IDS2017 dataset**, the Voting Classifier yielded the best overall performance, attaining the highest accuracy (0.95) and the lowest false positive rate (FP = 5180). The Autoencoder followed with a strong accuracy of 0.90 and an FP count of 62895. Isolation Forest lagged behind in both accuracy (0.26) and false positive performance (FP = 553795).

These results demonstrate that, in unsupervised scenarios, the Autoencoder and Voting Classifier consistently outperform

Isolation Forest across both datasets. The Voting Classifier, in particular, proves highly effective on CIC-IDS2017, while the Autoencoder shows a balanced performance on both datasets.

V. Conclusions

Our main goal was to reduce the number of false alarms. Based on the results presented, we can draw the following conclusions about performance in various training scenarios.

XGBoost outperforms other models, especially when training on a full dataset or on normal activity. Its tunability and ability to handle complex function interactions make it a good choice. However, this method has by far the most false positives for the first dataset, which plays a key role in our case. The second limiting factor in using XGBoost is the need for labeled data, which is very rare in real-world use

Autoencoders are competitive in terms of accuracy, with a much lower rate of false positives, but their effectiveness depends on proper architecture design and hyperparameter tuning, which takes time. In the examples studied, the results obtained by Autoencoders were better than those obtained by Isolation Forest. Autoencoders are particularly effective at anomaly detection because they learn to compress and reconstruct data in a way that best represents typical, frequent patterns. When trained on normal data, the model reconstructs well only those cases that are close to the training set, while outliers (anomalies) result in a much higher reconstruction error. This allows autoencoders to detect abnormal events without having to manually flag anomalies, making them particularly useful for unsupervised applications.

Isolation Forest struggles with unbalanced data sets and performs worse when trained only on normal activity. This is particularly evident with the second data set. Research confirms that despite its effectiveness and simplicity, Isolation Forests have some limitations in anomaly detection. Reserch First of all, they assume that anomalies are rare and easy to isolate, which can lead to errors with complex, high-dimensional data, where the boundaries between normal and anomalous classes are less clear. In addition, the method can have difficulty detecting contextual anomalies that are abnormal only under specific conditions, rather than globally. Isolation forests are also sensitive to the scale of the data, so they require careful normalization, and their performance can deteriorate when the data contain large amounts of noise or are unevenly dispersed in feature space.

A voting classifier helps increase classification accuracy by combining the decisions of several different models, leading to better generalizability and reducing the risk of overfitting. This makes the system more resilient to the errors of single classifiers, since the final decision is based on a majority of votes. It also allows flexible combining of different types of models to take advantage of their individual strengths. A particularly positive effect observed was a significant reduction in false positives which is highly desirable in real-world applications.

Acknowledgment

The authors would like to thank experts for their appropriate and constructive suggestions to improve this template.

References

[1] N. Chowdhury and V. Gkioulos, "Cyber security training for critical infrastructure protection: A literature review," *Computer Science Review*, vol. 40, p. 100361, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013721000010

[2] A. J. Field, "French optical telegraphy, 1793-1855: Hardware, software, administration," *Technology and Culture*, vol. 35, no. 2, pp. 315–347, 1994. [Online]. Available: http://www.jstor.org/stable/3106304

[3] I. J. Kadhim, P. Premaratne, P. J. Vial, and B. Halloran, "Comprehensive survey of image steganography: Techniques, evaluations, and trends in future research," *Neurocomputing*, vol. 335, pp. 299–326, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218312591

[4] G. S. Smith, "Recognizing and preparing loss estimates from cyber-attacks," *Inf. Secur. J. A Glob. Perspect.*, vol. 12, no. 6, pp. 46–57, 2004.

[5] T. Yadav and A. M. Rao, "Technical aspects of cyber kill chain," in *Security in Computing and Communications: Third International Symposium, SSCC 2015, Kochi, India, August 10-13, 2015. Proceedings 3*. Springer, 2015, pp. 438–452.

[6] T. J. Holt and M. Kilger, "Examining willingness to attack critical infrastructure online and offline," *Crime & Delinquency*, vol. 58, no. 5, pp. 798–822, 2012.

[7] D. Kushner, "The real story of stuxnet," *ieee Spectrum*, vol. 50, no. 3, pp. 48–53, 2013.

[8] R. Khan, P. Maynard, K. McLaughlin, D. Laverty, and S. Sezer, "Threat analysis of blackenergy malware for synchrophasor based real-time control and monitoring in smart grid," in *4th International Symposium for ICS & SCADA Cyber Security Research 2016*. BCS, 2016, pp. 53–63.

[9] S. Mohurle and M. Patil, "A brief study of wannacry threat: Ransomware attack 2017," *International journal of advanced research in computer science*, vol. 8, no. 5, pp. 1938–1940, 2017.

[10] S. Y. A. Fayi, "What petya/notpetya ransomware is and what its remidiations are," in *Information technology-new generations: 15th international conference on information technology*. Springer, 2018, pp. 93–100.

[11] J. Beerman, D. Berent, Z. Falter, and S. Bhunia, "A review of colonial pipeline ransomware attack," in *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW)*. IEEE, 2023, pp. 8–15.

[12] R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar winds hack: In-depth analysis and countermeasures," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2021, pp. 1–7.

[13] S. Leppänen, S. Ahmed, and R. Granqvist, "Cyber security incident report—norsk hydro," *Procedia Economics and Finance*, vol. 11, 2019.

[14] J. Cervini, A. Rubin, and L. Watkins, "Don't drink the cyber: Extrapolating the possibilities of oldsmar's water treatment cyberattack," in *International conference on cyber warfare and security*, vol. 17, no. 1. Academic Conferences International Limited, 2022, pp. 19–25.

[15] M. R. Gauthama Raman, C. M. Ahmed, and A. Mathur, "Machine learning for intrusion detection in industrial control systems: Challenges and lessons from experimental evaluation," *Facebook AI research*, 2021. [Online]. Available: https://doi.org/10.1186/s42400-021-00095-5

[16] J. Lo, *Whitelisting for Cyber Security: What It Means for Consumers*. Public Interest Advocacy Centre, 2011.

[17] N. Pureti, "Firewalls explained: The first line of defense in cybersecurity," *Revista de Inteligencia Artificial en Medicina*, vol. 15, no. 1, pp. 60–86, 2024.

[18] M. Plachkinova and K. Knapp, "Least privilege across people, process, and technology: Endpoint security framework," *Journal of Computer Information Systems*, vol. 63, no. 5, pp. 1153–1165, 2023.

[19] K. Thakur, M. L. Ali, S. Kopecky, A. Kamruzzaman, and L. Tao, "Connectivity, traffic flow and applied statistics in cyber security," in *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. IEEE, 2016, pp. 295–300.

[20] J. Bharadiya, "Machine learning in cybersecurity: Techniques and challenges," *European Journal of Technology*, vol. 7, no. 2, pp. 1–14, 2023.

[21] G. A. Pimenta Rodrigues, R. de Oliveira Albuquerque, F. E. Gomes de Deus, R. T. de Sousa Jr, G. A. de Oliveira Júnior, L. J. Garcia Villalba, and T.-H. Kim, "Cybersecurity and network forensics: Analysis of malicious traffic towards a honeynet with deep packet inspection," *Applied Sciences*, vol. 7, no. 10, p. 1082, 2017.

[22] M. Sourour, B. Adel, and A. Tarek, "Environmental awareness intrusion detection and prevention system toward reducing false positives and false negatives," in *2009 IEEE Symposium on Computational Intelligence in Cyber Security*. IEEE, 2009, pp. 107–114.

[23] Z. Pelletier and M. Uryasev, Abualkibash, "Evaluating the cic ids-2017 dataset using machine learning methods and creating multiple predictive models in the statistical computing language r," *International Research Journal of Advanced Engineering and Science*, 2020. [Online]. Available: https://irjaes.com/wp-content/uploads/2020/10/IRJAES-V5N2P184Y20.pdf

[24] J. Jinsi, "Deep learning algorithms for intrusion detection systems in internet of things using cic-ids 2017 dataset," *International Journal of Electrical and Computer Engineering*, 2023. [Online]. Available: https://www.researchgate.net/publication/367762160_Deep_learning_algorithms_for_intrusion_detection_systems_in_internet_of_things_using_CIC-IDS_2017_dataset

[25] A. Yulianto, N. Suwestika, and P. Sukarno, "Improving adaboost-based intrusion detection system (ids) performance on cic ids 2017 dataset," *Journal of Physics Conference Series*, 2019. [Online]. Available: https://www.researchgate.net/publication/333169769_Improving_AdaBoost-based_Intrusion_Detection_System_IDS_Performance_on_CIC_IDS_2017_Dataset

[26] O. O. Oluwakemi and U. A. Muhammad, "Comparative evaluation of machine learning algorithms for intrusion detection," *Asian Journal of Research in Computer Science*, 2023. [Online]. Available: https://www.researchgate.net/publication/374056114_Comparative_Evaluation_of_Machine_Learning_Algorithms_for_Intrusion_Detection

[27] B. Ingre and A. Yadav, "Performance analysis of nsl-kdd dataset using ann," in *2015 International Conference on Signal Processing and Communication Engineering Systems*, 2015, pp. 92–96. [Online]. Available: https://ieeexplore.ieee.org/document/7058223

[28] E. Gbashi and B. Mohammed, "Intrusion detection system for nsl-kdd dataset based on deep learning and recursive feature elimination," *Engineering and Technology Journal*, vol. Vol. 39 No. 7 (2021): Engineering & Science Issue /, 09 2021. [Online]. Available: https://www.researchgate.net/publication/354523827_Intrusion_Detection_System_for_NSL-KDD_Dataset_Based_on_Deep_Learning_and_Recursive_Feature_Elimination

[29] K. Al Jallad, M. Aljnidi, and M. S. Desouki, "Anomaly detection optimization using big data and deep learning to reduce false-positive," *Journal of Big Data*, vol. 7, pp. 1–12, 2020.

[30] R. da Silveira Lopes, J. C. Duarte, and R. R. Goldschmidt, "False positive identification in intrusion detection using xai," *IEEE Latin America Transactions*, vol. 21, no. 6, pp. 745–751, 2023.