Enhancing Network Security through Machine Learning: A Study on Intrusion Detection Systems

Filippo Ciliberti
Universita' di Bologna - Campus di Cesena
Digital Transformation Management
filippo.ciliberti@studio.unibo.it
Cesena, 04/10/2024

***Abstract*—This research aims to analyze the possibility to apply different machine learning approaches in order to enhance the effectiveness and reliability of the network intrusion detection. To check the performance of the proposed approach with conventional methods, it is applied to detect both normal and anomalous flows extracted from a real military network environment, a data set obtained from kaggle. Upon importing the dataset, the distribution and nature of the features where examined through an exploratory data analysis (EDA) phase. Visualizing the dataset helped me understand better how to elaborate on the project, as well as handling outliers, or creating new attributes. A variety of supervised learning techniques were explored and evaluated. The outcomes of this study would be used in proposing an Intrusion Detection System that is effective and can be used to improve cybersecurity in the organizational networks.**

1. Introduction

The world is more interconnected than ever. Today's organizations are exposed to a constantly growing number of network based threats that jeopardize the confidentiality, integrity and availability of their assets.

To cope with threats and menaces, SIEM systems record logs and events, so as to report any kind of illegal activity within a network. These information come from Intrusion Detection Systems that are software based application that detect anomalous and illegitimate behaviors, trigger alerts, allow for fast recovery and increase the cybersecurity resilience of organizations.

Nowadays, threats are enhanced in terms of complexity and adopt very sophisticated techniques. Traditional security measures are therefore outdated, as well as IDS.

In this research I explored my datasets before to improve my knowledge and address what I though was the most appropriate elaboration according the goal. Data visualization helped me in identifying patterns and trends and managing any data missing or null values to keep data consistency. I also discussed categorical elements that, while critical, had numerous zero values, which might disrupt the model's learning process. I found and kept outliers since they usually reflect anomalies important for intrusion detection.

I applied both feature engineering and label encoding to finetune the dataset. Converting categorical features into numeric values for machine learning processing became especially necessary due to label encoding. I then devised a correlation matrix to study the relationships between the remaining multiple features, helping me to avoid duplicate details in my model.

When the dataset was ready for use after cleaning and optimization, I split 80% for training and 20% for testing.

Finally, this research is based on the application of several machine learning approaches to improve IDS efficiency and effectiveness. Supervised learning, is very promising in enhancing the surveillance of network intrusions by analyzing patterns and estimate both normal or anomalous traffic, therefore threats.

The training and testing of multiple machine learning techniques—including Logistic Regression, Support Vector Machines, Decision Trees, Random Forest, AdaBoost, and Neural Networks—were part of the modeling for the research, designed for the classification of network connections.

To determine how effectively they can discover anomalies, these models were reviewed against a test dataset. The purpose of this study is to identify the best models to enhance the detection performance of IDS and boost organizational network defenses.

A key objective of the research is to find the ideal machine learning techniques for execution, while also comparing different traffic patterns and characteristics that could indicate potential and specific attacks.

2. Related Work

Understanding the goals of my project and subjectively understanding the kind of data I owned led me to start researching related works and datasets found on the internet. I discovered that the large majority of publicly available datasets categorize network traffic according to advance defined attack types, frequently labeled in the dataset itself. This style was not in tune with the path I planned to travel. As a result, I selected to persevere with my own strategy and to greatly understand the features within my dataset.

An especially useful resource in this process was the KDD Cup 1999 dataset which, even though it was different from mine, resembled it in many aspects. The dataset presented in KDD Cup 1999 helped me to clarify the definitions of a variety of attributes that appeared in both datasets. This phase was key to fully understanding the core meaning of the features and their relation to multiple traits of network traffic.

Given this advanced insight, I focused to the effective understanding and labeling of the features, with the goal of preparing the data correctly for supervised machine learning. While concentrating on the features themselves instead of using standard labels for attack types, I endeavored to develop a model that could discern unusual network behavior from the data relationships, making my method more flexible and suitable for diverse scenarios.

3. Proposed Method

Traditionally, IDS have been categorized in signatures and anomaly detection based. Signature-based intrusion detection systems checks new traffic against established attack patterns (signatures) which are constantly updated against a database of signatures, whereas anomaly-based systems identify uniqueness in behavior by comparing it ideal behaviors. While anomaly-based systems are more versatile, they still deal with the problem of high false positive rates. This was also took in consideration in choosing the best model for this system. Indeed, the goal is to create an anomaly based system that minimize the false positives but offers a robust approach to identify normal or anomalous traffic, so as to make IDS stronger.

The training of models on datasets enables them to identify difference between normal and abnormal behavior, including in dynamic and difficult network ecosystems. Using this strategy reduces our dependence on manual rule definition and allows us to identify fresh and changing threats.

Having understood the category in which my IDS falls into, I began my project.

3.1. Choice of Dataset

The choice of my data, was driven by the need of gaining useful datasets on the publicy available databases. On Kaggle, I found the "Network Intrusion Detection" dataset that provides analysis on a simulated LAN which constitutes an emulation of the military network environment of the US Air Force.

The dataset is composed of "Train.csv" and "Test.csv" which contain several connections to the network towards the understanding of wether is is "normal" or an "anomalous" connection.

The reference to the dataset is at the following link: Network Intrusion Detection dataset.

3.2. Importing the needed libraries

To perform data analysis and in developing the model and to evaluate the model performance I required the following libraries. "Pandas" was employed in data manipulation and cleaning since the features found in the dataset are structured. "Matplotlib" and "seaborn" enabled me to visually explain the distribution and the trends of the data set and make sense of data. To handle data in an efficient manner particularly arrays and data processing the 'Numpy' supported a large number of mathematical operations.

Scikit learn coded and evaluated models of logistic regression, decision tree classification, support vector classification, random forest, Ada boost and multi-layer perceptron classifier for completing work in classification.

Due to my evaluation, I was able to gauge the performance of the models in addition to determining their accuracy rate, precision, and recall; additionally, I identified their confusion matrix to determine the efficiency of the models.

3.3. Exploratory Data Analysis

The dataset applied in this study has 25,192 entries and 42 features, each highlighting several aspects of network traffic. Such features encompasses a combination of numerical and categorical data, starting from protocol types and service flags to source and destination byte counts, and also includes the number of access 'logged-in' 'error rates.

The train.info() outlined the data structure, inclusive of column names, data types, and any missing values. All columns in the dataset are complete, with no missing entries, and consist of three primary data types: int64, float64, and object. Data must be encoded before applying any sort of machine learning.

Further exploratory investigation was done using train.describe(include='object'). I was able to yield descriptive statistics for the categorical features, understanding the different protocol types, services, and connection flags, that populated the dataset. The network traffic types were various.

Aside from that, features including protocol-type, service, and flag allow for insights into the kind of network connections being investigated, and numerical features like 'src-bytes' and 'dst-bytes' provide important information about data transfer.

This first investigation demonstrated that the dataset is properly set up, featuring no absent values and a suitable combination of numerical and categorical information. The multiplicity of connection-specific features, along with error rates and traffic types, produces a rich framework for crafting machine learning models to identify anomalies and malicious behavior in the network.

3.4. Understanding Attributes

Crucial was to understand the attributes of the data, and how they could have weighted differently on the classification of the network traffic. The table, attached to the "ipynb" file, extensively explains each feature and its role in the network capture.

In addition, for what concerns the attribute "flag", there is a "Flag Code Table" in the same document which describes each flag and its role in the connection.

3.5. Visualizing data

The attributes of the dataset were investigated within the exploratory analysis to understand network traffic distribution. An important step in this was to analyze connection flags, protocols, services, and the fraction of anomalies in the data.

The flags identified a mixture of network activities, and the lion's share, representing the 59.4% of the data, were marked as 'SF' (normal establishment and termination). Important additional flags included 'S0' (27.82 %), showing failed connections, and 'REJ' (8.80 per cent), which represents invalidated connections. Using less common flags, namely 'RSTR', 'SH', and 'RSTO', together contributed smaller fractions, reflecting a variety of incomplete or abnormal connections. This flag analysis provides important understanding of the blend of conventional and irregular network actions present in the dataset.

When we analyze protocol usage, TCP dominated, making up 81.48 % of network traffic, and UDP accounted for 11.95 % and ICMP comprised 6.57 %. For several network environments, the extensive dependence on TCP is regular, pointing out its function in administering reliable, connection-oriented traffic. The availability of UDP and ICMP traffic points to the variety of communication protocols throughout the network.

The services incorporated in the network showed further patterns. The 31.77 % of the traffic was attributed to HTTP, which demonstrates elevated web activity. The services that were critical included (17.27 %) private along with domain-u (7.22 %), SMTP (5.75 %), and FTP data (5.54 %), demonstrating various standard and private protocols, especially in file transfer and email services.

Finally, the dataset presented a large number of anomalies, as 46.61 % of connections were deemed anomalous. The large fraction of irregular activity points out the important demand for effective intrusion detection systems that can spot and answer to possible security dangers.

This study gives a thorough perspective on the composition of network traffic, which sets the stage for the machine learning models that will serve to reveal intrusions.

3.6. Handling outliers... or not!

To find potential outliers in the given dataset, I calculated the Z-scores for each numerical feature using the following formula:

$$Z = \frac{X - \mu}{\sigma}$$

where X indicates the feature value, μ represents the feature's mean, and σ its standard deviation.

Z-score measures the statistical distance between a data point and the mean of a dataset. It's expressed in terms of standard deviations. Usually, a Z-score exceeding 3 signals reveals an outlier.

After applying this rule, I spotted a large number of outliers throughout various features. In any case, I decided against removing or taking care of these outliers when preprocessing my data. In many cases, outliers reveal abnormal behavior in the network, which could be critical for finding anomalies or irregular patterns vital for intrusion detection. Due to the critical focus on detecting anomalous activity in this study, I decided to keep these outliers, as they could be important to identifying potential security threats.

3.7. Feature Engineering

The principal aim during my feature engineering was to improve the dataset by removing less relevant features for the analysis. Identification of features with many zero entries possibly indicated either redundancy or a limitation in their utility for classification. Sorting these features according to the %age of zeros revealed which columns might provide less to the overall model performance.

As a result, I have included all the feature in a list, considered significant for my models. In this "feature-list" I have included features such as connection flags, error rates, types of service, together with other network properties, which were considered critical for the identification of network patterns.

The list of important features, once finalized, was benchmarked against the entire suite of available features in the dataset. Features not part of this refined list were the candidates to be removed. The excluded columns usually led to a high %age of zero values or were assessed to be of less importance for detecting abnormal behavior.

By "pruning" the dataset it became optimized for modeling and boosted efficiency by eliminating unnecessary noise and complexity.

3.8. Attack Table Creation

In the feature selection process, I created subsets of the attributes I found more relevant, in order to distinguish three distinct attack types: DoS/SYN Flood, Data Exfiltration, and Port Scanning. For these attacks, the most suitable features aligned with their associated network traffic behavior. My approach provided the flexibility to designs the feature

set for various attack types, which improves the labeling efficiency and boosts the accuracy of the training models.

As an illustration, the selection of features such as 'serror-rate', 'srv-serror-rate', and 'same-srv-rate' came about due to their direct relevance to connection errors and service request patterns characteristic of DoS/SYN Flood attacks. Also, Data Exfiltration attacks associated with the features service, 'dst-bytes', and 'srv-count' highlight data transfer behavior and possibly indicate unusual data movement. In the end, 'protocol-type', count, and 'rerror-rate' were chosen as features in Port Scanning to monitor the organized probing signatures typical of this kind of attack.

The intent was to improve the dataset by associating each attack type with its most important characteristics to enhance the computer learning model training stage.

3.9. Label Encoding

In the transformation process, I adopted two significant techniques to prepare the dataset for machine learning: logarithmic transformation and label encoding.

To address the skewness and so asymmetric distribution, a logarithmic transformation was applied to certain attributes to avoid that the machine learning models underperform. The log transformation helps to reduce this skewness by compressing large values and expanding small values, leading to a more balanced distribution.

The features that underwent log transformation include:

- **duration**: the length of the connection
- **src_bytes**: data bytes from source to destination
- **dst_bytes**: data bytes from destination to source
- **count**: number of connections to the same host
- **srv_count**: number of connections to the same service
- **same_srv_rate**: %age of connections to the same service
- **dst_host_same_srv_rate**: %age of connections to the same service at the destination host

These changes put the distributions of the features into common standard so that machine learning models were able to understand the fundamental patterns of the features.

In addition to the logarithm transformations, below the categorical values numerically encoded to meet modeling requirements:

Class Encoding: For classification, the target variable, class, was given binary codes: 1 for anomalies and 0 for normal traffic. A binary format proves to be perfect for classification tasks.

Flag Encoding: A mapping of integers was performed for the flag feature to reflect the status of a network connection (e.g., success or rejection).

Protocol Encoding: The protocol_type feature received encoding, applying numerical values to the most typical protocols found in the dataset, including TCP, UDP, and ICMP. A mapping of each protocol to an integer enabled discrimination between different communication techniques.

Service Encoding: Encoded on a predefined 'danger ranking', the service feature identifies the kind of network service accessed. This ranking gives numeric ratings to services consistent with their risk levels, placing those services ranked more vulnerable or having a higher risk accordingly.

3.10. Hold Out, 80 train, 20 test

The first step was to split the dataset into train and test datasets. The Stratified Split method played a crucial role in protecting the class distribution (normal versus anomalous connections) in both the training and testing sets. In imbalanced datasets specifically, this is an important technique, since it confirms that the model acquires knowledge from a balanced set of data.

There was a split in the dataset of 80% for training and 20% for testing.

4. Results

4.1. Model Training and Cross-Validation

To assess the performance of different machine learning models I used cross-validation since it strongly estimates how effectively a model will perform by splitting the training dataset into a number of subsets known as 'folds.' Specifically, I used 5-fold cross-validation, which works as follows: the training data set is broken apart into 5 equal sections. Through all iterations, a specific subset acts as the validation set and the model receives training from the additional 4 subsets. This project completes its course 5 times, where each subset acts as a validation set only one time. The final score regarding accuracy is determined after all iterations are performed. It was essential to reduce overfitting and ensuring that the model performs well across different sections of data, instead of a singles test split.

I trained six different models and for each of them I assessed the accuracy.

- **Logistic Regression Cross-Validation Accuracy**: 0.9547
- **Decision Tree Cross-Validation Accuracy**: 0.9961
-
- **SVM Cross-Validation Accuracy**: 0.9728
- **Random Forest Cross-Validation Accuracy**: 0.9970
- **AdaBoost Cross-Validation Accuracy**: 0.9816
- **ANN Cross-Validation Accuracy**: 0.9940

For my scope, I ultimately picked Random Forest and Artificial Neural Network (ANN) as the two strongest models for additional testing to complete the model selection. The decision was made because of both, their high cross-validation accuracies—99.70% recorded by Random Forest and 99.40% achieved by ANN. In the case of this IDS system, there are other aspects made these models especially suitable for intrusion detection in addition to their accuracy.

Random Forest is particularly effective when dealing with such complex datasets, as it might be a network traffic.

This ensemble learning method is very valid for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. This makes this model resistant to overfitting, critical for an intrusion detection system.

Artificial Neural Network, specifically the **MPL Classifier**, has pattern recognition capabilities. Indeed, ANN excel in capturing complex and non-linear patters and, its flexibility of learning from huge amount of data and the capability to generalize unseen data, made it my first choice.

While accuracy played an important role in choosing these models, their ability to handle complex structures, to handle over-fitting and give relevance to the most important features, justified my selection for the IDS.

4.2. Model Evaluation on Test Data

The aim was to assess how well the Random Forest and Artificial Neural Network (ANN) models did by using the test dataset to measure their generalizability and capability in finding anomalous network behavior. Both frameworks were learned from the training dataset and examined on the reserved 20% for testing. The results are based on key performance metrics: Accuracy, precision, recall and the confusion matrix collectively characterize the detailed performance of the models on data that is unseen.

Random Forest Classifier
The model demonstrated optimal performance on the test dataset:

- **Accuracy**: 99.70%
- **Precision**: 99.71%
- **Recall**: 99.69%
- **Confusion Matrix**:
  (2687 3)
  (12 2337)

The model signifies a substantial capability for correct identification of both normal and anomalous network connections at 99.70% accuracy. This points out that it succeeds just as well in eliminating false positives and finding real anomalies. The confusion matrix shows a handful of misclassifications: 3 false positives and a total of 12 false negatives from 5,039 samples, which highlight the stable personality and reliability of Random Forest. Owing to its talent for blending predictions from diverse decision trees, the model is particularly effective in classifying complicated traffic movements within networks, and is especially useful for intrusion detection.

Artificial Neural Network - MPL
The Artificial Neural Network (ANN), implemented as a Multilayer Perceptron (MLP), also achieved excellent results:

- **Epochs**: 300
- **Test Accuracy**: 0.9964
- **Loss**: 0.0162

The evaluation of ANN model over 300 epochs, revealed insightful results shown as well in the learning curve and accuracy graphs.

4.3. Model Performance Visualization

4.3.1. Learning Curve. From the learning curve, it means that the model was learning the data patterns well as expected. The curves levels off after more or less 50 epochs and the training loss (blue) fall off slowly, which implies that the model was becoming better at predicting the training set. In the figure below, the training loss (black) and the validation loss (red) curve indicates that the proposed model has not over-learned the training data and so is generalizing well with relatively small gap between both the Loss Curves. Instead, overfitting is negligible, and the model can easily generalize to new unseen data quite well. We can see at epoch 250 the small increase of the validation loss but it does not affect the overfitting of the model as the loss falls back to previous value on the next epochs.

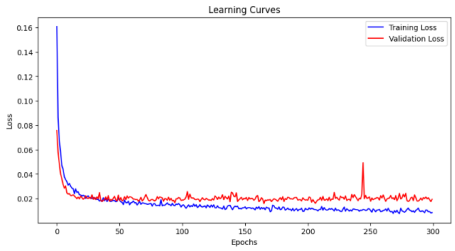

Figure 1. Learning curves

4.3.2. Accuracy Curve. Accuracy steadely increases for both training and validation sets, having a convergence of about 99.7%. This further confirms that the model is performing and generalizing well. The model does not overfit the training data as it can be seen approaching the close of the ages (200–300), the model shows stable and exceptionally high accuracy levels for validation (99.64%) and training (99.73%).

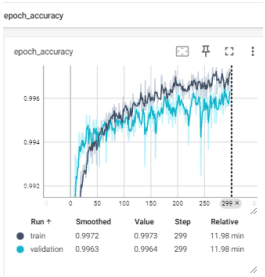

Figure 2. Accuracy over 300 epochs

5. Findings

These findings offer strong clarity on how these machine learning techniques are very effective for Intrusion Detection Systems (IDS). These models reliably and correctly identify anomalous network behavior, as shown by the 99.70% and 99.64% accuracies achieved, with minimal difference between training and validation sets.

For an IDS, this degree of performance is vital because the models have to effectively tell apart usual traffic from potential threats while maintaining a low error rate. The obtained high values show that the models generate very limited false positives, meaning that benign traffic is almost never misidentified as malicious, essential for sidestepping potential disruptions in network operations. In a like manner, the impressive recall values indicate that the models are very useful in recognizing true threats, guaranteeing that potential attacks are not missed.

This means, in real life, that the IDS is able to count on these models to monitor network activity in real-time and bring to attention any suspicious behavior with a strong degree of certainty. These models provide not only detection of existing attack patterns but also show strong generalization to new and unseen data, a significant requirement for adapting to changing threats. As a result, these results are not simply theoretical; they offer practical value in improving the security and resilience of networks.

5.1. Conclusion

This research leveraged machine learning models to improve the reliability of Intrusion Detection Systems, which nowadays are crucial for the defense of organizations' networks. After careful data preprocessing, data visualization and feature engineering, I aimed in building a system that could distinguish between normal or anomalous and security threats, thanks machine learning techniques.

Random Forest and Artificial Neural Network, emerged as to be the best fit for this IDS, achieving exceptional levels for the implementation in these kind of environments.

The model's ability to handle the complex interactions between various features in the network traffic made it an equally strong candidate for IDS deployment.

These results are not just theoretically significant; they offer real-world value. For an IDS to be effective, it must be able to reliably detect anomalies while minimizing false positives that can disrupt network operations. As shown for the two models tested, namely the Random Forest and the ANN the models showed a good ability to meet these requirements. High precision means that few normal traffic is mistaken as dangerous while high recall means that actual threats are not ignored. This balance if highly important when developing an IDS that should be able to function in dynamic environment, real-time environment where new threats are being developed every now and then.

Indeed, the overall performance boosting effect of the machine learning models under consideration has been proven for IDS in this particular study. Their high accuracy, precision and recall indicates that they are ready for deployment in current network environment to provide improved security from a number of threats. These models not only satisfy the current fundamental requirements for intrusion detection but are also flexible and extensible to construct more complex systems in the future of cybersecurity.

References

[1] Cost-based Modeling and Evaluation for Data Mining With Application to Fraud and Intrusion Detection: Results from the JAM Project by Salvatore J. Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K. Chan.

[2] Sampada Bhosale - Network Intrusion Detection

[3] Abhay Mudgal - Intrusion Detection System