Abstract

This thesis is about explainability and network intrusion detection systems, as we will see there aren’t works that match our criteria which are about making an explainable model by using the most recent datasets and keeping an eye on making something useful for the real world. To achieve that we will focus on the requirements of a recent work that highlights how most of the studies out there have flaws in developing a model that is useful outside of the academic world. On top of that, we will add some other requirements like testing the model with zero-day attacks, which in most of the related work is missing while is a crucial benchmark for a ml model. We will use Shap to make a true-to-the-model explanation to validate our model comparing the feature importance with how the actual attacks work. To the best of our knowledge, no other studies put together all these requirements making them less suitable for producing a model in a network intrusion detection system.

Introduction

Section 1 - Datasets used

Before starting with the paper review of the dataset a brief introduction to the subject is necessary. These datasets are made from raw data, in this case, the raw data are stored in files with PCAP format. These files contain all the network traffic of a given time seen by the capturing device, which is a network card. Once the single packets are stored flowmeters tools can build a dataset containing all the flows of the captured traffic. The flows are connections between devices, in particular, TCP flows have a packet to announce the beginning of the connection and another one to close the connection, but it can happen that the connection will be closed after a timeout. In contrast, UDP connections which can be preferred to TCP because of the lower overhead don’t contain in the standard a way to close the connection this task is demanded by the applications that are built on top of the UDP standard. These two standards work at level 4 of the ISO/OSI stack called the transport layer.

In February 2024 the three dataset-related papers with more citations in Google Scholars were CIC-IDS 2017 (Sharafaldin et al., 2018) with 3149 citations, NLS-KDD (Tavallaee et al., 2009) with 5131 citations and UNSW-NB15 (Moustafa & Slay, 2015) with 2740 citations which are improvements based on the raw data of KDD Cup ’99 (Lee et al., 1999). In this paper, we will focus on CIC-IDS 2017. CIC-IDS 2017 comes from CIC (Canadian Institute for Cybersecurity), IDS (Intrusion Detection System), and 2017 is the year. By making this dataset the authors have achieved the goal of covering all the 11 criteria given by (Gharib et al., 2016) that none of the already existing papers has achieved. This dataset has been made using CICFlowMeter (Lashkari et al., 2017) which is a software that can capture the network traffic or use already captured traffic files and make a csv file with all the flows on the network. They ran a test that lasted five days on Monday no attacks were performed, on Tuesday some brute force attacks were performed, on Wednesday a series of DoS attacks were performed, Thursday was an infiltration attack, Friday was Botnet, DDoS, and Port scan. For 12 attack labels, 1 benign label, and 80 traffic features were provided. The dataset is composed of multiple csv files for each time that they have run a different test, since we are interested in all the attacks, we just merged these datasets. However, as well as KDD Cup ’99 also CIC-IDS 2017 has received some improvement, thanks to the availability of the pcap files and the open-source nature of CIC-IDS 2017.

In (Engelen et al., 2021) and (L. Liu et al., 2022) the pcap files of CIC-IDS 2017 and CSE-CIC-IDS 2018 were analyzed to discover how the labeling process of both the datasets wasn’t accurate leading to some benign traffic categorized as attack and some attack labeled with another attack label. The level of corruption of the original dataset is over 5%, which means that every past paper about CIC-IDS 2017 has learned some wrong patterns including the ones with nearly perfect scores. This work has led to the creation of a new dataset called WTMC-2021. They have produced documentation of 160 pages on how to correctly label the attacks. They also have improved CICFlowMeter by fixing some issues related to the timeout, assigning the correct flow to packets with a payload of 0 bytes, correcting the calculation of existing attributes, and implementing new attributes. Moreover, they decided to modify the CICFlowMeter to close a connection not at the first FIN package but after two to follow the TCP standard (Brownlee et al., 1999). They also noticed that CICFlowMeter was ignoring the RST flag, which is supposed to close the TCP connection, so they have added this feature. Another improvement that has been made is to add new labels for the flows that are attempted attacks but without malign payload, these flows come from very long flows that are divided in two because of the connection timeout. These flows might have the attack payload only in one of the flow entries created by CICFlowMeter. They have also tested the impact on machine learning models and generally, they have seen an improvement in the scores thanks to the fact that each class has the right label.

(Lanvin et al., 2023) make another improvement over WTMC-2021 – they use only the improved version of CIC-IDS 2017 - called CRiSIS-2022, they found that the pcap files contain some duplicates within 500 microseconds and that they aren’t in the correct order, moreover, they also have added a label for port scan which was reported on the paper of CIC-IDS 2017 bus wasn’t there on the dataset. They make different tests to see how these improvements affect the metrics for different models and they discover that adding the Port Scan label and deleting the duplicated packets drop the TPR by 20%, from 100% to 80%. While by adding the Port Scan label and keeping the duplicated the TPR remains at 100%. It is clear how all the models based on the original CIC-IDS 2017 need to be run with this and further updates to have reliable scores.

The latest improvement is (Pekar & Jozsa, 2024) where they show how the improved CICFlowMeter still shows some issues, so they decided to go for NFStream (Aouini & Pekar, 2022). The first issue was related to the fact that there is an anomalous count of FIN and RST Flags which are the flags used by TCP for closing the connection, the problem is that CICFlowMeter continues to count the number of flags even after the connection is supposed to be dropped. The normal behavior is supposed to be according to the first fix of CICFlowMeter that at the second FIN flag the connection end, while at the first RST flag the connection end. However, they found that the connection remains open, leading to having all the statistics of the affected flows related to two or more flows instead of one. The second issue is that even after the patch CICFlowMeter still creates some flows with missing values or negative values in fields where only positive values are supposed to be. Their proposal is two datasets NFS-2023-nTE which close a connection after the timeout is reached, this dataset was created with the intent to be compared with WTMC-2021 and CRiSIS-2022 to see the differences between the NFStream version and the CICFlowMeter version. NFS-2023-TE closes the connection after the first FIN or RST flag like most of the flow analyzers do (Hofstede et al., 2014), making this dataset the closest to a real word dataset among them. Another difference of this dataset is that they decided to drop duplicates within 10000 packets instead of 500 microseconds like was done in CRiSIS-2022. The authors have tested their new datasets, and they did notice huge improvements in the score compared to WTMC-2021 and CRiSIS-2022, however, they confirmed that these two datasets make a huge improvement compared to CIC-IDS 2017. Before showing these results the authors talked about feature selection and they say that features such as IP addresses and ports have been removed, however, there is no mention of removing the timestamps from the data.

It’s important to mention that the author of the paper in an email exchange with the author of CICFlowMeter has discovered that a new Python-based tool is supposed to come out. We also talked with the authors of CICFlowMeter, we had a confirmation that the new tool will have more than 50 new features for a total of 130 features. The authors of CICFlowMeter have collected the criticality of the software and they are working on this improved version, this is important since having a tool that is a standard for the community can lead to training and testing the same model on different datasets.

Hikari 2021 (Ferriyan et al., 2021) is a dataset that is inspired by CIC-IDS 2017 but includes 7 new requirements, they provide as well the pcap files alongside a csv file and a pkl of the dataset. They provided 84 traffic features and 2 target features one multi-categorical and the other one binary, among the 84 traffic features most of them are inspired by the ones of CIC-IDS 2017. In this dataset, the authors have focused only on web attacks since 80% of the attacks nowadays are done on the application layer. So, they have performed 3 different attacks, but on the dataset, a fourth one is present since while they were analyzing the background traffic, they found a crypto miner attack. The raw data have been extracted using tcpdump, while the labeling and the analysis of the background traffic were made with Zeek which has also added some features that aren’t present in CIC-IDS 2017. These features are the IP address and ports of source and destination and uid. The usage of Zeek, which is a well well-known tool in the community, avoids the mentioned issues with CICFlowMeter but the authors didn’t provide any guidance on how they have used the tool and how they have made the csv file out of the Zeek logs. However, in an email exchange, the authors declared that they are working on the release of the code that they used to build the dataset and when asked about adding the feature timestamp they said that they are considering adding the feature in future updates of the dataset.

--- spostare l’analisi sotto

While waiting for future updates we have analyzed both the raw data and the dataset itself some duplicates and unsorted have been found but they aren’t enough to make a significant impact. Hikari-2021 has 0.2% packet out-of-order in CRiSIS-2022 which has the same amount makes a difference on all the selected metrics which is less than 0.1% given the nature of the problem is reasonable. While in CRiSIS-2022 the duplicated packet makes 20% of the difference in the TPR metrics, but it has 4,5% of duplicates which in Hikari-2021 is only 0.03% 150 times less so there should be an impact but is supposed to be less significant than the one seen in CRiSIS-2022.

Another issue found on HIKARI-2021 is that shows the same FIN and RST wrong count of WTMC-2021 and CRiSIS-2022, we can assume that this is caused by using a timeout for closing the connections instead of closing the connection at the first FIN or RST flag. As a reference NFS-2023-nTE there are 1.9% of packets with over 2 FIN flags counted, and 7.3% of packets after the first RST flags counted. While HIKARI-2021 has 0.5% observations over 2 FIN flags counted and 8.2% observations after the first RST flag counted.

Another analysis done in this study shows how the timeout wasn’t there at all, the biggest duration seen in a connection was about 17942.909297 seconds which is a little bit less than 5 hours which is the duration of the entire data capturing session that we can retrieve by analyzing the pcap files provided. The authors were asked about this issue we are waiting for an answer.

Before looking at the solutions that we proposed is important to say that the right way to address these issues is to build the dataset from scratch, but since is not possible because of the lack of documentation we decided to do some data engineering on the dataset.

To address these issues, we can limit the duration of a flow to 1800 seconds, like was done in CIC-IDS 2017 and NFS-2023-nTE. This is necessary to have a reasonable response time in case of an attack, the flaw here is that in CIC-IDS 2017 the timeout was about 120 seconds for idling connection and 1800 for the active ones, but we don’t know if the flow with a duration of over 1800 were idling or active. Related to the flows with over 2 FIN flags counted these will be dropped, while the ones with 2 flags counted will be changed to 1 flag counted and the related ACK flag counter will be decreased by 2. The idea is that since the professional flow maker tools stop at the first FIN flag seen we need to reproduce what would be the statistics for that case. So since without a timeout, we are supposed to see 2 FIN flags this number will be decreased to 1 and the ACK decreased by 2 since an ACK flag is supposed to be sent after a FIN flag. Although we know that this can be a limitation because it might happen that between the two FIN flags, some data can be transferred this is the only way to address this issue.

Hikari-2021 and NFS-2023-TE were chosen because they are some of the latest up-to-date datasets for Intrusion Detection at the time, this is quite important as they highlight in their papers that older datasets make use of older attacks so training a model on those datasets is no longer useful in the real world more on that in section 3. Moreover, the older datasets have also less than 80 features, making these two datasets more appealing. In addition, not only the raw data are useful for trusting these datasets and making overtime improvements, but they also allow the creation of new datasets as was done by (Kabla et al., 2022) where the raw data of Hikari and CTU-13 (García et al., 2014) have been merged.

Section 2 - Explainable algorithm

Shap (Lundberg & Lee, 2017) is a Python library that makes use of a game theoretic approach by computing an approximation of the Shapley values. Shapley values are used to fairly assign to each player in a game an additive value which shows how the player has contributed to the game. To do that all the possible combinations of players have to play the game and then by looking at the results with and without a given player the value of the player is computed. To do this in machine learning is needed that a model is trained with all the possible combinations of feature sets, but this is a very expensive process. Here Shap came to help, the library has different approximation explainers that allow us to compute an approximation of the Shapley values in a faster way than the original algorithm without the requirement of multiple training.

These explanations can be local or global, a local explanation is about explaining only one row of the dataset while a global is about explaining the feature importance of the entire dataset. To make a global explanation there are two alternatives, one is to use the absolute mean of all the observations, and the other one is to use the highest absolute value among the observations. The difference between these two alternatives is that the first one shows the average impact of a feature in the overall data, and the latter shows how the importance of infrequent but with high magnitude effect features.

In this work, we will talk about Kernel Shap which adapts the Lime (Ribeiro et al., 2016) algorithm to integrate the Shapley values. This algorithm makes a surrogate linear model to computer the Shapley values, these kinds of explainers are called model agnostic since they can run with all the machine learning models. One problem of Kernel Shap is that since it makes use of a linear model it may make wrong values when some features are highly correlated. Other than Kernel Shap another explainer has been introduced in that paper which is Deep Shap (Shrikumar et al., 2017) Deep Shap has adapted the algorithm of Deep Lift to include Shapley values and works only with neural networks, so a model-specific explainer. Lately, other two papers (Lundberg et al., 2018, 2020) introduced Tree Shap, which is another model-specific explainer for trees and most of the tree-based ensemble models.

Tree Shap explainers, as well as Linear Shap, can provide true-to-the-model explanations (Chen et al., 2020) as opposed to the other explainers that are true-to-the-data. By breaking the dependence between features following the rules of casual inference (Janzing, 2019) true-to-the-model explanations assign a value different from zero to a feature only if it’s used by the model. In the case of true-to-the-data explanation if two features correlate 100% the same importance will be assigned to them even if one is not used.

Before going further with an explanation is important to say what a perturbation is and what is background data. The perturbation means that the background data is used to randomly swap the value of random features in the data that we want to explain to see how these changes impact the result. The perturbation is used as a workaround to the fact that removing a feature from the data without training again the model is not feasible, so instead of seeing how the result changes by removing features swapping values is used. The consequences of this approach are that the more data we use as background data more the explanation will be accurate, usually between 100 and 1000 observations are recommended. In case there is any doubt about the dimension of background data test can be done by using the same amount of data while changing the data used to see if there is any difference in the results. As well different amounts of data can be used to see if there is any variation in the explanation. Another factor that can improve the accuracy of the explanation is the number of perturbations per observation, which is how many times I do the perturbation over the same observation. When it comes to global explanations another variable that plays an important role is the dimension of the sample, when a bigger sample can achieve better results.

Is important to notice that the values of the background data need to be sampled values of the dataset or centroids of k-means to be realistic samples. The reason is that if we assign a random number to a feature let’s say 1000 while the real range of this feature is from 0 to 2 shap will assign a very high importance to that feature even which is not realistic. Given that, for having true-to-the-model explanations only training data need to be used since the model has assigned the feature importance based on the data seen so far while a bigger or smaller range in the testing data leads to different feature importance which talks more about the dataset then the assumption used by the model.

In the example of a logistic regression, instead of using background data the weight of the coefficients can be used to explain the training data to obtain a true-to-the-model explanation. This is feasible since by already knowing the coefficient of a feature shap can understand what the consequence would have changing values to each feature. This is called interventional feature perturbation.

Instead, if we choose the correlation-dependent perturbation same importance will be given to highly correlated features, and background data is needed. In the case of Tree Shap, both the perturbation options make a true-to-the-model explanation, with the difference that interventional perturbation requires background data while tree path dependent will use the examples stored in the nodes of the tree which is the training dataset.

Section 3 - Requirements

In the survey (Zhang et al., 2022) the authors highlight some limitations of explainable AI for cyber security, which is a big subject that also includes intrusion detection. Some of the limitations that they highlighted as fields where some future work can be on are problems related to how to deal with privacy issues on the datasets, a trade-off between model performance and explainability, and how to deal with adversarial attacks.

In the specific case of Intrusion Detection Systems, the problem related to privacy is that research can’t record the traffic of a public network for making an example because sensitive data can be contained in the payload of the packets. To deal with this issue the authors of Hikari 2021 have anonymized specific parts of the traffic by using crypto pan.

Then there is the problem of the trade-off between the performance of the model and explainability, regarding this point, different solutions can be adopted. One could use a glass box model, like the explainable boosting machine (Lou et al., 2013) which is a GA2M model which is an explainable model while accurate like the state-of-the-art black box model. Another alternative is to use a post hoc explanation like shap, which in the case of ensemble models random forest, lightgbt, or XGBoost can explain the entire training set providing a true-to-the-model explanation.

Regarding adversarial attacks, they use the example of (Slack et al., 2020) in which they show how LIME and Kernel Shap can be fooled by generating a fake perturbation of a dataset while the model is a black box model that when a feature goes out of the distribution always gives the same outcome. In this paper, they make an example of a bank that wants to create a black box model to classify their customers, and they want to include some features like sex that aren’t supposed to be used. So, they show how is possible to make a fake model and how this model paired with some data generated ad hoc can give the wrong explanation. So that in case of an external audit, they can provide the black box model alongside the sample of generated data to fool the results. But to do that the model needs to be a black box model, while for using a model-specific algorithm like tree shap access to an actual model is needed. For example, when feature perturbation tree-path-dependent is used the explainer must see what each node of the tree has inside. So, unless both the model and the explainer have been replaced with a malicious one this is not an issue for us. This doesn’t mean that this solution is 100% guaranteed against all possible attacks, but at least the scenario described in that paper is not a real problem for model-specific explainers.

We have seen how some of the issues shown in the previous survey can be addressed, now it’s time to look at some issues that are specific to ML models in the context of NIDS. (Catillo et al., 2023) have written a paper about some of the most common issues encountered in machine learning models applied to network intrusion detection systems. They have started by criticizing typical issues of the datasets and then they have moved to the implications from the machine learning perspective. Since this work takes as an example CIC-IDS 2017 we will go issue by issue to see how the improvements of CIC-IDS 2017 and HIKARI-2021 have worked to deal with those issues.

The authors of HIKARI-2021 in an email exchange state that are available to address the issues mentioned here, we are waiting for an answer with the required details.

Simplification of the data collection environment

As far as we know, no dataset has achieved this requirement which consists of running a network with different workloads, topology, and traffic conditions.

Contemporaneity and effectiveness of the attack

All the datasets based on CIC-IDS 2017 fail to meet this requirement since they are now 7 years old, and they have little in common with modern attacks. Moreover, the DDoS attacks of the dataset are not effective as discussed in (Catillo et al., 2021).

However, in Hikari 2021 the authors decided to focus only on encrypted traffic at the application layer since nowadays 80% of the attacks are done at the application layer which is the 7th level of the ISO/OSI stack, making this dataset as far as we know the best option to meet these requirements.

Representativeness of the normal baselines

Both these datasets fail to meet this requirement since CIC-IDS 2017 only contains in its baseline four protocols, while Hikari 2021 only HTTPS traffic. If the model sees only certain protocols when some normal traffic of another protocol is detected the model can think that this is attack traffic or the other way if it sees only attacks from certain protocols some attack traffic from a new protocol can be classified as normal.

Bugs of the feature extractor and incorrect ﬂow records

As we have seen NFS-2023-TE relies on NFStream which is a tool with a wide user base, so the feature extractor tool is not supposed to be an issue. Moreover, before use the raw data needs to be sorted and deduplicated and this work has been done in CRiSIS-2022 and the following improvements.

Hikari makes use of Zeek which is another tool used by professionals that was used – at the time was called Bro - in KDD 99’ and UNSW-NB15. Besides Zeek another tool was used but, in the paper, they don’t mention which tool is nor do they provide the code, they just mention a generic Python tool. In an email exchange, they declared that they are planning to release the script, so this flaw will be addressed in the future. For the moment how Zeek was used and how they have made the dataset out of the Zeek log files is not known, which is a problem for the accountability of the dataset that we will face later in Section 5.

--- spostare questa parte sotto

However, as reported in Section 1 we found that the connection ended with a timeout instead of using the appropriate flags, this is not a bug of Zeek but a misconfiguration of the tool. Related to the raw data we found 0.2% of unsorted data, which isn’t supposed to be a problem as shown in CRiSIS-2022, and 0.03% of duplicates which is a problem but given the small percentage the impact shouldn’t be huge.

Data Labeling

This problem might happen when the background traffic is not validated and labels are applied only by looking at the timestamp, source, and destination. This is not the case for Hikari 2021 and the improvements of CIC-IDS 2017. Related to Hikari 2021 the authors in an email exchange have said that they validated with Zeek both benign and background traffic while doing this an attack was found and labeled while validating background traffic.

Class imbalance

|  |  |  |  |
| --- | --- | --- | --- |
| **Label** | **count** | **%\_over\_majority\_class** | **%\_over\_all** |
| Benign | 562335 | 100 | 71,7693365 |
| Background | 170151 | 30,25794233 | 21,71592445 |
| Probing | 23388 | 4,159086665 | 2,984948904 |
| XMRIGCC CryptoMiner | 10874 | 1,933722781 | 1,38782001 |
| Bruteforce-XML | 8795 | 1,564014333 | 1,12248271 |
| Bruteforce | 7988 | 1,420505571 | 1,019487423 |

**Table 1** Hikari class imbalance analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Label** | **count** | **%\_over\_majority\_class** | **%\_over\_all** |
| BENIGN | 2639806 | 100 | 83,71407007 |
| PortScan | 231572 | 8,772311299 | 7,343658827 |
| DoS Hulk | 158546 | 6,005971651 | 5,027843316 |
| DDoS | 95685 | 3,624698179 | 3,034382373 |
| FTP-Patator | 7963 | 0,301650955 | 0,252524292 |
| DoS GoldenEye | 7917 | 0,299908402 | 0,25106553 |
| DoS slowloris | 5192 | 0,19668112 | 0,16464977 |
| SSH-Patator | 2980 | 0,112887083 | 0,094502372 |
| DoS Slowhttptest | 2732 | 0,103492454 | 0,086637745 |
| Bot | 738 | 0,0279566 | 0,023403608 |
| Web Attack - Brute Force | 151 | 0,005720117 | 0,004788543 |
| Infiltration | 28 | 0,001060684 | 0,000887942 |
| Web Attack – XSS | 27 | 0,001022802 | 0,00085623 |
| Web Attack - Sql Injection | 12 | 0,000454579 | 0,000380546 |
| Heartbleed | 11 | 0,000416697 | 0,000348834 |

**Table 2** NFS-2023-TE class imbalance analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Label** | **count** | **%\_over\_majority\_class** | **%\_over\_all** |
| BENIGN | 1612267 | 100 | 76,36982262 |
| PortScan | 223886 | 13,88640963 | 10,60502641 |
| DoS Hulk | 158027 | 9,801540316 | 7,485418953 |
| DDoS | 93178 | 5,779315709 | 4,413653156 |
| DoS GoldenEye | 7916 | 0,490985674 | 0,374964888 |
| DoS slowloris | 5192 | 0,322031028 | 0,245934525 |
| FTP-Patator | 3992 | 0,247601669 | 0,189092955 |
| SSH-Patator | 2980 | 0,184832909 | 0,141156565 |
| DoS Slowhttptest | 2727 | 0,169140719 | 0,129172467 |
| Bot | 738 | 0,045774056 | 0,034957565 |
| Web Attack - Brute Force | 151 | 0,009365694 | 0,007152564 |
| Infiltration | 27 | 0,001674661 | 0,001278935 |
| Web Attack – XSS | 27 | 0,001674661 | 0,001278935 |
| Web Attack - Sql Injection | 12 | 0,000744294 | 0,000568416 |
| Heartbleed | 11 | 0,000682269 | 0,000521048 |

**Table 3** NFS-2023-nTE class imbalance analysis

We can see in Table 2 and Table 3 that as well as the original CIC-IDS 2017 his latest improvement has a problem with class imbalance for example Heartbleed has only 11 observations which in both datasets is 0,0006% or less. We can notice in Table 1 that Hikari 2021 with the 2022 update of the dataset has all its classes over 1,4% and Bruteforce which is the smallest one has 7988 observations which is enough to train a model even if we decide to do undersampling on the other classes to 7988 observations.

After having seen all the issues from the dataset side, is time to focus on the ML implications.

Attack-revealing features and ease of detection

As we will see in the next section it’s not rare to find papers that make use of all features in the dataset which is an issue since some of them are revealing. In the case of this kind of dataset, an IP address can be enough to reach a very high score if the IP is one of the attackers. Another mistake that is reported in the next section is to include the index of the dataset, which for this kind of data is revealing since within certain ranges of indexes almost all the flows rely on an attack. Moreover, some datasets even when revealing features are removed show patterns that make it easier even for a OneR model (D’hooge et al., 2023), to achieve a high score.

Data partitioning

Two problems were discussed, the first one is shuffling or sampling some data which is supposed to belong to an arbitrary timestamped order in the case of the use of a deep learning model with memory.

Another problem of random sampling is when you have a small minority class that is not present in the training dataset because was randomly excluded. For this work, we decided to use the stratified under-sampling technique, as well as the stratified split for the testing and training dataset to always have some samples of the minority class in the dataset.

Unmotivated complexity

The issue reported in the paper is that deep networks require a lot of resources and time and often the authors of the papers that make use of these technologies don’t discuss what’s the footprint of their work and the time required to train such big models. Moreover, to prove that a giant deep network makes sense a fair comparison with another model that can be for example a random forest is needed.

On top of that if we add explainability as a requirement as discussed in section 2 a complex model mean that the post hoc explainer will be very expansive in term of computing time and less accurate.

Use of the evaluation metrics

When it comes to evaluation metrics is important that the metrics are complete and appropriate for the given task. Is not unusual to see papers that show an accuracy of 99% on a dataset with 90% of positive labels without making use of other metrics like for example f1 score.

Since for this thesis, random stratified undersampling has been chosen for Hikari, and a hybrid approach that makes use of both over-sampling and undersampling has been chosen for NFS-2023-nTE and NFS-2023-TE this is not a concern for the study. The goal of this sampling strategy is to have all the classes with the same amount of data.

Lack of transferability

For the moment being able to create a network intrusion detection system with a public dataset and being able to run in the real world by keeping the same results is still an open problem.

Section 4 – Related work

Now we propose an analysis of other papers that aim to build a machine-learning model with the selected dataset to see how they addressed the problems and if they contain flaws. We selected all the papers that have used Hikari 2021 flows for making ml models which are 6, the top 4 most cited papers about CIC-IDS 2017 and the top 2 most cited papers about CIC-IDS 2017 that made use of Shap. This selection is not broad because while reading and reporting the score is a relatively fast task, criticizing the paper while trying to understand the difference in every aspect of the methodology is not. Hower is important to understand every aspect of the methodology since some papers say that the random forest is the best model ever, while others say the opposite, but the reality is that the sampling of the data was done differently leading to opposite outcomes.

The first flaw that they have in common is that none of them have published the source code of the work, which can help to understand how things have been done when they are not explicitly mentioned in the paper. For example, if I want to make a comparison with a paper that doesn’t provide for example f1 score I can run the model and add these metrics to the used metrics.

None of the selected papers about Hikari 2021 have used the updated version of the dataset, this is a big flaw since the latest update not only adds new flows but also addresses the problem of class imbalance. This improvement comes from another run of the test bed that was done one year after the first one. Moreover none of those papers have noticed the anomaly of having too many FIN and RST flag counted on the dataset.

Talking about CIC-IDS 2017 there aren’t related works that make use of the last improvement of the dataset, however, is more important to find a paper with a solid methodology so that in case it can replicated with the improved dataset.

The paper that has heavily inspired this work is (Kwon et al., 2023) where they used Hikari 2021 and tested different models such as Random Forest, XGBoost, MLP, and CNN. One limitation of this study is that only the MLP and the CNN have been tuned, while the Random Forest as well as the XGBoost were not. They opted to fine-tune the MLP with CIC-IDS 2017 to be able to make a comparison with other papers already available and then to use the same parameters with Hikari 2021. Even if these two datasets share some similarities, not only the data but also the features are not the same. In the paper, they have shown how doing random under-sampling can improve the selected metrics which are the f1 score and detection rate. Then they make a test to see if the models can address zero day attacks (Ali et al., 2022) which are unseen attacks that were only present in the test set. Being able to detect this kind of attack is one of the reasons why machine learning is used over signature-based IDs that can detect only known attacks. To test if the model can discover unseen attacks, they excluded one by one each attack and then they tried to predict the background traffic as well as the removed attack to see how each model performed. They have found that only the MLP and CNN were able to detect the brute force XML and probing when missing in the training data.

Then we have (Noori et al., 2023) who talk about feature selection, but they don’t mention which features have been selected in the first place before doing feature selection, which for a paper that highlights how it is useful and makes a comparison with different off methods is a downfall. This is for example one of the cases where providing the source code is useful so that we can look at the used features and see what features have been used. Then while they talk about the attacks, they mention only three attacks out of four while they promote the use of SMOTE for doing oversampling on the minority class, when the smallest class has been removed. No explanation has been given about why this feature has been removed. Then they reference to figure 4a which is supposed to contain a pie chart with the attack distribution of Hikari 2021 but the attacks in the figure aren’t the ones of Hikari 2021. Moving to Figure 7 only 2 out of the 4 attacks were used and no explanation is provided. Making a study related to a specific attack can make sense, but if some attacks were removed without giving any explanation might be a mistake, or even worse might be done to increase the score in the metrics.

(Louk & Tama, 2023) is the latest paper available related to our research but it shows some issues, first, they don’t tell which features were used during the training and if the prediction is binary or multi-class. Then they said that the code was available, but no link was provided, and they didn’t reach out after an email request, so it wasn’t possible to understand which features were used. Moreover, when we tried to see how the model performed, we got their best model which is a bagging of 50 gradient boost machines with 500 trees a model that predicts everything as not an attack. So, because of the lack of transparency, this paper wasn’t helpful for our work. Again, providing the source code could be the definitive solution to help reproduce this paper.

(Rajak et al., 2022) has proposed a model made with a CNN combined with an LSTM but as they show in Table 2 during the process of feature selection, they have chosen both the traffic category and the Label, so one of the two possible target labels was on the training dataset.

In (Fernandes & Lopes, 2022) and (Fernandes et al., 2023) the author makes use of the features Unnamed: 0.1 and Unnamed: 0 which are not actual features. Unnamed: 0.1 is created because Pandas by default assumes that the first column of a CSV file is a feature but in this case is an index. While Unnamed: 0 was already there, again is not a legit feature. Moreover, HIKARI-2021 has a pkl version where these columns aren’t present meaning that one of the columns mentioned in these studies has been introduced by the authors.

The problem with these columns is that since the attacks were run sequentially a Random Forest which is used on the paper can learn the number of rows where the attack begins and stops.

In (Chauhan & Shah Heydari, 2020) Shap has been used for feature selection which in (Marcilio & Eler, 2020) has been proved to be a good alternative to other feature selection techniques such as RFE, mutual information, and ANOVA. However, in this study, they use GAN (Goodfellow et al., 2014) which creates new data used for training the model and to do that they opted to select only DDoS attacks. Moreover, no explanation about why to use only a DDoS attack, which has no practical uses since the attacks used in CIC IDS 2017 can be avoided just by setting in the right way the server (Catillo et al., 2021).

Then we have (Sarhan et al., 2022) which use only datasets that provide the raw data to make a comparison between three different datasets using both NetFlow and CICFlowMeter they show how NetFlow-based datasets can achieve better scores on the select metrics. But have already seen that the issue is not the features used by CICFlowMeter, but the bugs in the tool itself making this comparison useless. Moreover, since the work of making a new dataset with NetFlow was done with the original CICFlowMeter also the labels are not right. That said the author didn’t mention that the CICFlowMeter version of the dataset has some features with a correlation that goes over 90%, so even if we assume that the dataset used is the latest improvement of CIC-IDS 2017 to make a fair comparison some prior feature selection needs to be done.

(Zavrak & Iskefiyeli, 2020) uses Auto Encoder, Variation Auto Encoder, and One-Class SVM these models are unsupervised models that to be trained need to see only normal traffic while during the testing normal traffic and the attacks were provided. For evaluating the results, the AUC was used, but they didn’t provide a weighted mean of the results, given that they have kept the dataset unbalanced these results are meaningless.

In (Kurniabudi et al., 2020) feature selection using Information Gain on CICIDS-2017 has shown some improvement, they also have made a comparison with NSL-KDD to show how some features that are present in CICIDS-2017 and not in NSL-KDD are useful to detect some attacks. However, one limitation of this study is that only 20% of the dataset was used. During their analysis of the dataset, they also found one redundant feature. Other than that, they also did a relabel following the work of (Panigrahi & Borah, 2018) where they reduced the number of labels by creating a grouping of Brute Force attacks, DDoS attacks, and Web Attacks. In this way, the new classes are less imbalanced, but there are some issues with that: first, they haven’t proved that the grouped attack behaves in the same way, and second, the smallest attack in this dataset is Heartbleed which has only 11 observations. So, this relabeling doesn’t prevent all these observations from falling either in the training or testing data. In fact, after running their model they found that an Infiltration attack that only had 36 observations wasn’t detected by their model, while for Heartbleed nothing has been reported since it was part of the DDoS class.

To address the problem of imbalanced learning (Yulianto et al., 2019) have used AdaBoost with smote, showing how the model with smote and feature reduction was able to achieve better results with the over-sampled dataset. They also have done feature selection selecting 15 features that have performed better than 10 components of PCA with 95% of the variance. However, these results as well as the previous paper aren’t suitable for the real world since they have used only DDoS attacks.

(Maseer et al., 2021) is another of the most cited papers about CIC-IDS 2017, they have tested 48 combinations of model and hyperparameters but reported only 31 since the others were showing poor results. They tried to face the problem of training AIDS which is one of the models chosen with multi class citing other studies that have used this model only for binary classification. As well as the previously mentioned datasets they reduce the number of classes to 4, choosing the ones present in the Thursday web attack file which are Brute Force, XSS, and SQL injection.

(J. Liu et al., 2021) has made a comparison between SMOTE and ADASYN while training a LightGBM model, they have made use of the entire dataset as well as NSL-KDD and UNSW-NB15. They also faced the problem of the smallest class by doing stratified training and test datasets. However, they didn’t use as metrics only accuracy and false alarm rate even if other metrics like precision and recall are cited in the methodology. So, without these metrics or even more complex metrics like AUC, f1 score or MCC is not possible to understand if the high accuracy is given by the unbalanced dataset or because the model works.

Section 5 - Methodology used

This work will follow were possible the requirements discussed by (Catillo et al., 2023) with the addition of a zero-day attacks test, model explanation, and the release of the Python code on Git Hub. All the chosen models rely on the boosting technique since these models can perform well even with an unbalanced dataset, and they can provide true-to-the-model explanations. Related to the datasets random stratified undersampling has been performed, and

Ali, S., Rehman, S. U., Imran, A., Adeem, G., Iqbal, Z., & Kim, K. Il. (2022). Comparative Evaluation of AI-Based Techniques for Zero-Day Attacks Detection. *Electronics (Switzerland)*, *11*(23). https://doi.org/10.3390/electronics11233934

Aouini, Z., & Pekar, A. (2022). NFStream: A flexible network data analysis framework. *Computer Networks*, *204*. https://doi.org/10.1016/j.comnet.2021.108719

Brownlee, N., Mills, C., & Ruth, G. (1999). *Traffic Flow Measurement: Architecture*.

Catillo, M., Pecchia, A., Rak, M., & Villano, U. (2021). Demystifying the role of public intrusion datasets: A replication study of DoS network traffic data. *Computers & Security*, *108*, 102341. https://doi.org/10.1016/j.cose.2021.102341

Catillo, M., Pecchia, A., & Villano, U. (2023). Machine Learning on Public Intrusion Datasets: Academic Hype or Concrete Advances in NIDS? *Proceedings - 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume, DSN-S 2023*, 132–136. https://doi.org/10.1109/DSN-S58398.2023.00038

Chauhan, R., & Shah Heydari, S. (2020). *Polymorphic Adversarial DDoS attack on IDS using GAN*.

Chen, H., Janizek, J. D., Lundberg, S., & Lee, S.-I. (2020). *True to the Model or True to the Data?* https://github.com/slundberg/shap/blob/

D’hooge, L., Verkerken, M., Wauters, T., Volckaert, B., & De Turck, F. (2023). *Castles Built on Sand: Observations from Classifying Academic Cybersecurity Datasets with Minimalist Methods*. https://orcid.org/0000-0001-5086-6361

Engelen, G., Rimmer, V., & Joosen, W. (2021). Troubleshooting an Intrusion Detection Dataset: The CICIDS2017 Case Study. *Proceedings - 2021 IEEE Symposium on Security and Privacy Workshops, SPW 2021*, 7–12. https://doi.org/10.1109/SPW53761.2021.00009

Fernandes, R., & Lopes, N. (2022). Network Intrusion Detection Packet Classification with the HIKARI-2021 Dataset: a study on ML Algorithms. *10th International Symposium on Digital Forensics and Security, ISDFS 2022*, 1–5. https://doi.org/10.1109/ISDFS55398.2022.9800807

Fernandes, R., Silva, J., Ribeiro, O., Portela, I., & Lopes, N. (2023). The impact of identifiable features in ML Classification algorithms with the HIKARI-2021 Dataset. *ISDFS 2023 - 11th International Symposium on Digital Forensics and Security*, 1–5. https://doi.org/10.1109/ISDFS58141.2023.10131864

Ferriyan, A., Thamrin, A. H., Takeda, K., & Murai, J. (2021). Generating network intrusion detection dataset based on real and encrypted synthetic attack traffic. *Applied Sciences (Switzerland)*, *11*(17). https://doi.org/10.3390/app11177868

García, S., Grill, M., Stiborek, J., & Zunino, A. (2014). *An empirical comparison of botnet detection methods*. https://doi.org/10.1016/j.cose.2014.05.011

Gharib, A., Sharafaldin, I., Habibi, A., & Ghorbani, A. (2016). *An Evaluation Framework for Intrusion Detection Dataset*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). *Generative Adversarial Nets*. http://www.github.com/goodfeli/adversarial

Hofstede, R., Čeleda, P., Trammell, B., Drago, I., Sadre, R., Sperotto, A., & Pras, A. (2014). Flow monitoring explained: From packet capture to data analysis with NetFlow and IPFIX. *IEEE Communications Surveys and Tutorials*, *16*(4), 2037–2064. https://doi.org/10.1109/COMST.2014.2321898

Janzing, D. (2019). *Causal Regularization*.

Kabla, A. H. H., Thamrin, A. H., Anbar, M., Manickam, S., & Karuppayah, S. (2022). PeerAmbush: Multi-Layer Perceptron to Detect Peer-to-Peer Botnet. *Symmetry*, *14*(12). https://doi.org/10.3390/sym14122483

Kurniabudi, Stiawan, D., Darmawijoyo, Bin Idris, M. Y. Bin, Bamhdi, A. M., & Budiarto, R. (2020). CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection. *IEEE Access*, *8*, 132911–132921. https://doi.org/10.1109/ACCESS.2020.3009843

Kwon, D., Neagu, R. M., Rasakonda, P., Ryu, J. T., & Kim, J. (2023). Evaluating Unbalanced Network Data for Attack Detection. *SNTA 2023 - Proceedings of the 2023 on Systems and Network Telemetry and Analytics*, 23–26. https://doi.org/10.1145/3589012.3594898

Lanvin, M., Gimenez, P. F., Han, Y., Majorczyk, F., Mé, L., & Totel, É. (2023). Errors in the CICIDS2017 Dataset and the Significant Differences in Detection Performances It Makes. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13857 LNCS*, 18–33. https://doi.org/10.1007/978-3-031-31108-6\_2

Lashkari, A. H., Gil, G. D., Mamun, M. S. I., & Ghorbani, A. A. (2017). Characterization of tor traffic using time based features. *ICISSP 2017 - Proceedings of the 3rd International Conference on Information Systems Security and Privacy*, *2017-Janua*, 253–262. https://doi.org/10.5220/0006105602530262

Lee, W., Stolfo, S. J., & Mok, K. W. (1999). *Mining in a Data-flow Environment: Experience in Network Intrusion Detection*.

Liu, J., Gao, Y., & Hu, F. (2021). A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM. *Computers and Security*, *106*, 102289. https://doi.org/10.1016/j.cose.2021.102289

Liu, L., Engelen, G., Lynar, T., Essam, D., & Joosen, W. (2022). Error Prevalence in NIDS datasets: A Case Study on CIC-IDS-2017 and CSE-CIC-IDS-2018. *2022 IEEE Conference on Communications and Network Security, CNS 2022*, 254–263. https://doi.org/10.1109/CNS56114.2022.9947235

Lou, Y., Caruana, R., Gehrke, J., & Hooker, G. (2013). Accurate intelligible models with pairwise interactions. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *Part F1288*, 623–631. https://doi.org/10.1145/2487575.2487579

Louk, M. H. L., & Tama, B. A. (2023). Dual-IDS: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. *Expert Systems with Applications*, *213*, 119030. https://doi.org/10.1016/j.eswa.2022.119030

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S. I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, *2*(1), 56–67. https://doi.org/10.1038/s42256-019-0138-9

Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2018). *Consistent Individualized Feature Attribution for Tree Ensembles*. http://github.com/slundberg/shap

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, *2017-Decem*, 4766–4775. https://github.com/slundberg/shap

Marcilio, W. E., & Eler, D. M. (2020). From explanations to feature selection: Assessing SHAP values as feature selection mechanism. *Proceedings - 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2020*, 340–347. https://doi.org/10.1109/SIBGRAPI51738.2020.00053

Maseer, Z. K., Yusof, R., Bahaman, N., Mostafa, S. A., & Foozy, C. F. M. (2021). Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. *IEEE Access*, *9*, 22351–22370. https://doi.org/10.1109/ACCESS.2021.3056614

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings*, 1–6. https://doi.org/10.1109/MilCIS.2015.7348942

Noori, M. S., Sahbudin, R. K. Z., Sali, A., & Hashim, F. (2023). Feature Drift Aware for Intrusion Detection System using Developed Variable Length Particle Swarm Optimization in Data Stream. *IEEE Access*, *11*(September), 1–1. https://doi.org/10.1109/access.2023.3333000

Panigrahi, R., & Borah, S. (2018). A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *International Journal of Engineering and Technology(UAE)*, *7*(3.24 Special Issue 24), 479–482. https://www.researchgate.net/publication/329045441

Pekar, A., & Jozsa, R. (2024). *Evaluating ML-Based Anomaly Detection Across Datasets of Varied Integrity: A Case Study*. https://intrusion-detection.distrinet-research.be/WTMC2021/tools

Rajak, P., Lachure, J., & Doriya, R. (2022). CNN-LSTM-based IDS on Precision Farming for IIoT data. *Proceedings of 4th International Conference on Cybernetics, Cognition and Machine Learning Applications, ICCCMLA 2022*, 99–103. https://doi.org/10.1109/ICCCMLA56841.2022.9988997

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). *“Why Should I Trust You?” Explaining the Predictions of Any Classifier*. https://doi.org/10.1145/2939672.2939778

Sarhan, M., Layeghy, S., & Portmann, M. (2022). Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-Based Network Intrusion Detection. *Big Data Research*, *30*, 100359. https://doi.org/10.1016/j.bdr.2022.100359

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy*, *2018-Janua*(Cic), 108–116. https://doi.org/10.5220/0006639801080116

Shrikumar, A., Greenside, P., Shcherbina, A. Y., & Kundaje, A. (2017). *Not Just A Black Box: Learning Important Features Through Propagating Activation Differences*. https://arxiv.org/

Slack, D., Hilgard, S., Jia, E., Singh, S., & Lakkaraju, H. (2020). Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 180–186. https://doi.org/10.1145/3375627.3375830

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, *Cisda*, 1–6. https://doi.org/10.1109/CISDA.2009.5356528

Yulianto, A., Sukarno, P., & Suwastika, N. A. (2019). Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset. *Journal of Physics: Conference Series*, *1192*(1). https://doi.org/10.1088/1742-6596/1192/1/012018

Zavrak, S., & Iskefiyeli, M. (2020). Anomaly-Based Intrusion Detection from Network Flow Features Using Variational Autoencoder. *IEEE Access*, *8*, 108346–108358. https://doi.org/10.1109/ACCESS.2020.3001350

Zhang, Z., Hamadi, H. Al, Damiani, E., Yeun, C. Y., & Taher, F. (2022). Explainable Artificial Intelligence Applications in Cyber Security: State-of-the-Art in Research. *IEEE Access*, *10*(July), 93104–93139. https://doi.org/10.1109/ACCESS.2022.3204051