2021 13th International Conference on Cyber Conflict
Going Viral
T. Jančárková, L. Lindström, G. Visky, P. Zotz (Eds.)
2021 © NATO CCDCOE Publications, Tallinn

Explainable AI for Classifying Devices on the Internet

Artūrs Lavrenovs
NATO CCDCOE
Tallinn, Estonia
arturs.lavrenovs@ccdcoe.org

Roman Graf
Accenture GmbH
Vienna, Austria
roman.graf@accenture.com

Abstract: Devices reachable on the Internet pose varying levels of risk to their owners and the wider public, depending on their role and functionality, which can be considered their class. Discussing the security implications of these devices without knowing their classes is impractical. There are multiple AI methods to solve the challenge of classifying devices. Since the number of significant features in device HTTP response was determined to be low in the existing word-embedding neural network, we elected to employ an alternative method of Naive Bayes classification. The Naive Bayes method demonstrated high accuracy, but we recognise the need to explain classification results to improve classification accuracy.

The black-box implementation of Artificial Neural Networks has been a serious concern when evaluating the classification results produced in most fields. While devices on the Internet have historically been classified manually or using trivial fingerprinting to match major vendors, these are not feasible anymore because of an ever-increasing variety of devices on the Internet. In the last few years, device classification using Neural Networks has emerged as a new research direction. These research results often claim high accuracy through the validation employed, but through random sampling there always occur devices that cannot be easily classified, that an expert intuitively would classify differently. Addressing this issue is critical for establishing trust in classification results and can be achieved by employing explainable AI.

To better understand the models for classifying devices reachable on the Internet and to improve classification accuracy, we developed a novel explainable AI method, which returns the features that are most significant for classification decisions. We employed a Local Interpretable Model-Agnostic Explanations (LIME) framework to

explain Naive Bayes model classification results, and using this method were able to further improve accuracy with a better understanding of the results.

Keywords: *classifying devices on the Internet, machine learning, explainable AI, Naive Bayes*

1. INTRODUCTION

With billions of devices connected to the Internet, it is not a question of whether the devices will be compromised or abused but when. Hundreds of millions of devices that are publicly reachable on the Internet are particularly vulnerable. Unsophisticated attackers can freely communicate with these devices and exploit configuration weaknesses or unpatched publicly disclosed vulnerabilities. Even if there are no currently disclosed vulnerabilities, these can appear at any time in the future. The classification of devices on the Internet has emerged in the last few years as an important research topic in the context of cyber security. Researchers and defenders have to understand new threats quickly and precisely in order to respond swiftly. Longstanding issues have to be understood so as to identify and address the root causes.

What class of devices has been compromised to create the latest Internet of things (IoT) botnet? What classes of devices have been abused for decades for distributed denial-of-service (DDoS) attacks? What devices receive a few anomalous network traffic flows from our network? These are just a few of many questions researchers and cyber security professionals have to answer. A pattern in the questions can already be observed inquiring about either large sets of devices or a few individual ones. Traditionally, this has been addressed either by applying a limited set of static classification rules or manual investigation by an expert. The increase in the number of devices is accompanied also by an increase in their heterogeneity. Static rules cannot keep up with this trend; therefore, the precision and also suitability of this method is decreasing. Expert availability is limited, and time is valuable – the automation of expert knowledge is the holy grail of AI application in the cyber security domain.

Expert knowledge, especially in cyber security, stretches far beyond applying standard tools and techniques. An expert's intuition is built upon years of experience, and the ability to validate predictions. An expert understands that a cloud computing network should not have many ICS devices present on it. And the few that might have a purpose would be specific to the infrastructure of the data centre. In comparison, even a sophisticated ML classifier classifies a large number of IoT devices in commercial

hosting and cloud networks [1]. Is this only an issue of lacking a network name or type (e.g. residential, commercial, cloud) as a feature? This feature can easily be added [2], but it is still not a guaranteed fix. While this is an obvious anomaly that could be easily identified and addressed by an expert, it is unclear how prevalent the misclassification is in the current research body, which stems from the black-box approach of the ML classification.

Feature selection is generally based on expert knowledge. Experts attempt to transfer their knowledge and characterise their intuition in the specific domain similar to how static rules would be created. Features for device classification with HTTP interfaces include directly identifiable keywords (the typical way to define a static rule), the behaviour of the responses, exclusion conditions, and the statistical properties of the responses [3], [4]. Expert feature decisions can be based on external sources of information; for example, a detailed scan of the device, which makes validating classification results much harder. In all the cases, the validity of feature selection can be questioned. While it is unfeasible for the expert to define all the less common features, experts can easily miss relationships between common features or put too much emphasis on some. Without explainable classification, feature selection and tuning can become overly reliant on the initial expert input.

The trustworthiness of classification even before adversary attempts are considered is the main obstacle to adoption in production. While the continuous improvement of the classification of identified issues and increasing precision is the expected progression in research, a single misclassification can be catastrophic in a cyber defence setting. Another advantage of the suitability of this research is that the level of precision is approaching expert knowledge for large sets, which has never before been possible.

The contributions of this paper include applying explainable AI to the problem of the classification of devices for the first time in published literature and bridging the gap between expert knowledge and automated classification.

Section 2 reviews related work, and Section 3 describes the application of explainable AI for the problem of device classification. Section 4 analyses the classification explanation for a random device from each defined class. Section 5 provides an overview of the overall classification results, and Section 6 provides final conclusions.

2. RELATED WORK

Scanning the Internet for specific devices or protocols is an established practice in security research. This type of research in itself has no novelty in respect to the classification process. Assumptions can be made that a device with a known open port corresponding to a non-generic protocol is serving a role that could be easily classified. Further validation by executing protocol communications can be conducted and data potentially useful for classification extracted. This methodology is effective for locating high-impact devices that are running specific protocols (commonly ICS) for the purpose of disabling public access. Mirian *et al*. scanned the Internet for common industrial protocols while identifying the discrepancy between open ports and the ability to handle respective protocol handshakes [5]. Dahlmanns *et al*. explore the security issues for the publicly reachable industrial protocol OPC UA [6]. Feng *et al*. automated IoT classification rule generation [7].

The privileged observer can identify traffic passing through network routers. The basic properties of port and protocol communication can be similar, while active communication requires sophisticated fingerprinting. This approach might make it possible to identify devices that are not publicly reachable but are actively communicating, while at the same time, it might miss devices that are not actively sending packets. Nawrocki *et al*. utilised IXP and ISP vantage points to identify common industrial protocols while still being challenged by traffic classification [8].

The research into AI classification consists of the same two vantage point approaches. The main challenge is identifying features and labelling sufficient training sets. Yang *et al*. identified and classified ICS and IoT devices extracting features and fingerprints from multiple communication layers [9]. Augmenting this with automated rule generation saved a significant amount of work for labelling the training set. Lavrenovs *et al*. trained classifier targeting interfaces based on generic HTTP protocols [2]. Privileged network observer classifiers are commonly trained on labelled data either from a laboratory network [10] or a campus network [11], [12]. Yadav *et al*. provide a systematic categorisation of ML augmented techniques for fingerprinting IoT devices [13].

Due to the fact that many AI models follow the black-box approach in terms of result transparency, research in the explainable AI domain has evolved drastically in recent years. Multiple frameworks such as Local Interpretable Model-Agnostic Explanations (LIME) [14] and SHapley Additive exPlanation (SHAP) [15] have been developed, aiming to facilitate the implementation of AI in different domains, by providing transparency and trust in underlying models. Different explainable AI solutions are already employed in the IoT domain. An IoT system [16] of low-cost

sensors incorporates an explainable AI decision support system. Another IoT system [17] makes use of an approach within the human-centric AI field for generating explanations about the knowledge learned by a neural network (in particular a multilayer perceptron) from IoT environments. We selected the LIME framework for our implementation, as it is one of the most robust and established solutions.

3. AI FOR DEVICE CLASSIFICATION

The input for classification is a scanning output – HTTP responses in a JSON format. The established text classification methods often suffer from large vector sizes and are less effective as the number of samples rises. The most effective method is a neural network [18], which learns automatically from examples, but suffers from a lack of results transparency. We are addressing this drawback using explainable AI. Another effective method for particular IoT use cases is Naive Bayes [19], which often serves as a robust method for data classification, but vectors representing an incident in Naive Bayes are larger than in the word-embedding methods of the neural network approach. However, in the case of IoT devices, we have experimentally identified that data is sparse, and the vector size is not large. A Naive Bayes method expects each feature in an HTTP response to be independent of all other features. Consequently, for the particular use case of classifying IoT devices, we suggest using Naive Bayes for text classification.

A. Features Used for Classification

We rely on features of HTTP responses suitable for the classification that have previously been developed and described in detail in [2]–[4]. These include HTTP response headers and the respective values, network name, HTML tree structure hash, body title, body keywords, SSL certificate issuer, and subject.

B. Data Sets

The primary data set consists of Internet scans of web interfaces. These scans are created by tools commonly used for Internet research – zmap and zgrab2 [20]. Both the HTTP default port 80 and the common alternative port 8080 were scanned in December 2020. Up to three redirects were followed to any port including HTTPS, in which case TLS negotiation was also saved. This toolset makes it possible to acquire research data in a uniform way, where zmap conducts Internet-wide (IPv4 only) scanning for open TCP ports in an optimised manner and hands over the identified services to zgrab2 for communicating on the HTTP application level, extracting response properties (headers, body, TLS, encountered errors), and formatting in a suitable way for further processing.

For the standard port, there were 51,118,537 elements, and for the alternative port, 8,343,898 elements. An element is a response (and appropriate redirects) corresponding to a single request that contains at least one proper HTTP response. We have augmented the elements in the data sets with additional features. The network name was looked up via the Maxmind GeoIP database. HTML tree hash, first title and body words were all generated from the response HTML body itself.

Secondary data sets (for 2018 and 2019) were utilised only to provide a comparison of the classification differences using the newly proposed Naive Bayes application, and the results are presented in Section 5. These older data sets are analysed in detail in [2].

We rely on the labelled set consisting of 171,791 elements developed in [2]. This was created from random elements of the 2018 port 80 data set, and therefore is unbalanced across classes. There are 132,562 WEB, 22,002 NET, 9,561 IPCAM, 711 INFRA, 265 VOIP, 243 ICS, 218 IOT, 153 PRINTER, 4,175 UNCLEAR and 1,901 UNCATEGORIZED devices in this labelled set. Class motivations and definitions are described in detail in [2]. WEB devices are generic web sites. ICS devices serve some industrial purpose and thus might be the most impactful class. NET devices provide network connectivity to both residential and large-scale networks. IPCAM provide networked video surveillance or recording. INFRA devices provide infrastructure functionality for virtualised and related services. VOIP devices provide IP telephony services. IOT devices include all IoT and smart home products. PRINTER class consists of printers and printing servers. It is impossible to determine the class of UNCATEGORIZED devices while UNCLEAR are likely embedded devices without a clear role but not serving a WEB role.

C. Comparison with the Neural Network Classification

To classify IoT devices, we examined two AI models. The first model [2] was the Neural Network (NN) with Word Embeddings, which provided good results with high classification accuracy (87%). But the drawback of this model is that classification results are difficult to explain due to the black-box description of the NN structure. The second model is a Naive Bayes (NB) classification model, which is fast and easy to implement.

The Multinomial NB is often used for document classification problems, using the frequency of the words existing in the document as input for the calculation. The main difference between the NB model and the NN model is that the predictors are regarded as independent. Examining features extracted from IoT device responses, we concluded that they should not necessarily be regarded as dependent, because they come from different independent response parts such as the header, and different

body parts, which describe independent aspects such as the domain, company name, technologies. The position of a word within a sentence and possible relations between the words can be omitted in the case of sparse data in an IoT device response. Therefore, we can assume the independence of IoT features and employ the NB model for classification. Additionally, as described in a comparative NN vs NB study [21], NNs have a long training time and require a large number of parameters that are best determined empirically. It has been observed that the NB classifier outperforms ANN learning algorithms in all cases. NB is a generative model, which assumes conditional independence, as in the IoT devices case, whereas NNs are discriminative models, which only model the probability of the class given the input, as described in [22] and [23].

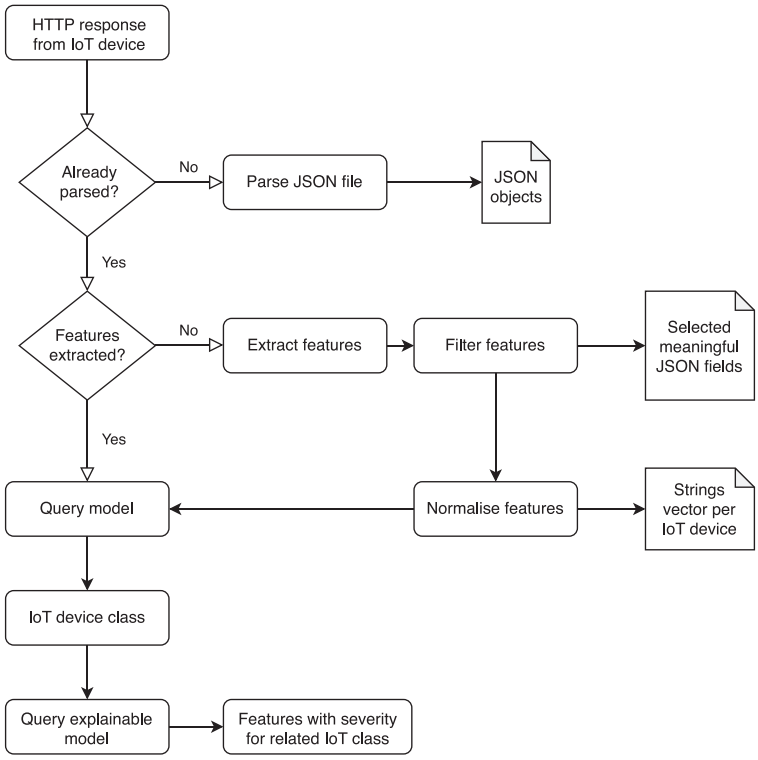
Another consideration supporting the NB model was that for meaningful NN training, long feature vectors are better, whereas IoT device feature vectors are often quite short and NB models could perform better in this particular case. The NB model is also a better match to explain it afterwards using the LIME [24] model. The LIME model explains the predictions of NB classifiers, providing rational numbers and associated features as text, which allows the human interpreter to understand if the feature word was negative or positive for each word in the IoT device response. With the LIME model, we aim to understand specific predictions to investigate the NB model whenever we doubt a given classification.

D. General IoT Device Classification Workflow

Device classification employs feature extraction and training of the NB model for queries. Classification predicts previously defined categories for a given sample. There are ten expert-defined classes: ICS, INFRA, IOT, IPCAM, NET, PRINTER, UNCATEGORIZED, UNCLEAR, VOIP and WEB. Supervised learning employs labelled training data to learn mapping functions from a given input (list of words) to the desired output value (class name). The workflow process is composed of two parts. One process is NB model training, where the workflow acquires device data from different sources such as the Internet and domain experts. The model is trained and regularly updated using extended knowledge from new device crawls. Figure 1 provides an overview of device classification using NB. This approach is based on a knowledge base containing a large number of labelled responses in JSON format (Step 1). This data can be provided by different means, collected at different times for particular operating systems, and can be separated by type of application and protocol. The novelty of this approach is that, for typical use cases, we propose to have associated decision rules for initial labelling. All such rules are then aggregated in a common labelled dataset, which supports final classification. We send requests to devices, and the system extracts features (Step 2) from the response and stores them for further analysis and queries the model that was trained on the knowledge

base. During the feature extraction, we execute parsing, filtering, and normalising of the content. The final classification result is based on querying the model (Step 3) or cache if the sample hash is already known and is a report in the form of a particular class name. To explain classification results, we query an explainable model (Step 4) based on the NB model and receive features with their severity for a particular class.

FIGURE 1: THE WORKFLOW FOR FEATURE EXTRACTION, DEVICE CLASSIFICATION, AND CLASSIFICATION EXPLANATION USING A NAIVE BAYES



Having an HTTP response from IoT devices in the form of a JSON file, we can classify the given IoT device description to one of the earlier defined IoT device classes employing the Naive Bayes algorithm (1). This formula shows the probability of the IoT device description D (2) belonging to the IoT device class c. The probability of the IoT device description D is a product of all specifications ds that are comprised in the IoT device vocabulary.

$$p(c|D) = \frac{p(D|c)p(c)}{p(D)} \quad (1)$$

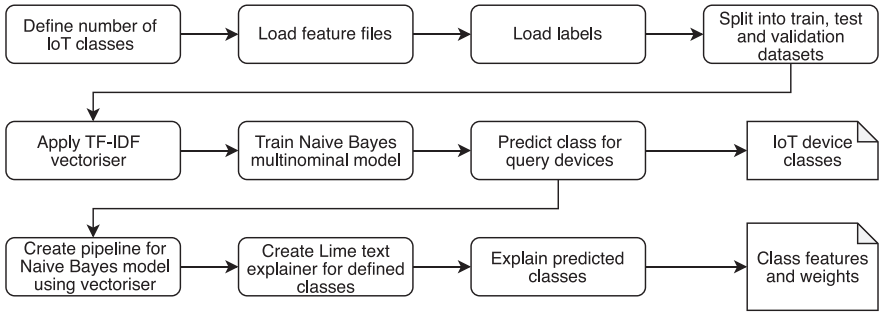
$$D = (ds_1, ds_2 \ldots ds_{10}) \quad (2)$$

The Naive Bayes algorithm picks the IoT device class with the highest probability and reports this as a classification result.

E. Naive Bayes and Explainable Model Training

The data for NB model training is prepared as described in Figure 1 in the previous section. We start with the definition of IoT classes. In the next step, we load a labelled dataset, dividing it into train, test and validation datasets. After acquisition and feature extraction, the input for the model is a list of words for each sample. For the detection of the most valuable features, we apply a TF-IDF vectoriser. TF-IDF helps to exclude words that are too frequent. For tokenising, filtering and normalising features, we filter out common and stop words, remove punctuation and special characters, remove non-alphanumeric characters, convert to lower case to have case insensitive matching, and normalise size. In the tokenising step, we break down each sentence to a set of single words. This is then converted into the one-hot vector to be processed at the input level of the NB model in Figure 2. To perform training, features aggregated in text form must be converted into numerical values, since machine learning algorithms cannot process plain text. Therefore, each uploaded sample (see Figure 2) is converted into an array of strings, where each string represents a particular feature. Then strings are encoded using indices, and each feature string has a unique index. If this feature repeats in the samples, we re-use its index. Finally, arrays of indexes are converted to one-hot encoded vectors, meaning that the position of each feature in the original feature set is encoded using "1" if a feature exists in the given place or "0" if not. The NB training and accuracy calculation process took 15.723163 seconds. To explain classification results, we create an explainable model by creating a pipeline for the previously calculated NB model, using a vectoriser. Using the vectoriser we create a LIME text explainer for the classes defined in the first step of the workflow. Finally, using the LIME text explainer, we explain the classes predicted by the NB model and obtain related class features and weights for each query sample. The LIME text explainer calculation took 3.345 seconds. Each query takes approximately 1 second for the whole workflow.

FIGURE 2: THE WORKFLOW FOR NAIVE BAYES AND EXPLAINABLE MODEL TRAINING



We trained a balanced model to avoid the bias in the original large labelled dataset, because we identified that by randomly sampling the classified output of the whole data set the small model performed better. As the full labelled data set primarily consists of WEB devices, the classified output is significantly skewed towards classifying devices as WEB. To avoid the bias of overrepresented classes in the labelled data set (in total 171,791), such as WEB, we employ a balanced labelled training set (in total 11,479): ICS:243, INFRA:711, IOT:218, IPCAM:1,999, NET:2,000, PRINTER:153, UNCATEGORIZED:1,901, UNCLEAR:1,999, VOIP:265, WEB:1,999. The labelled training data set was divided into a training set (5,628), validation set (2,413), and test set (3,447). The test accuracy is 82%. Therefore, the classification results can be interpreted by humans able to reason and explain why a certain classification was derived. We can acquire the explanations for different features in a numerical form, meaning their weights with positive and negative signs, which means that words that are weighted negative towards one class may be positive towards another.

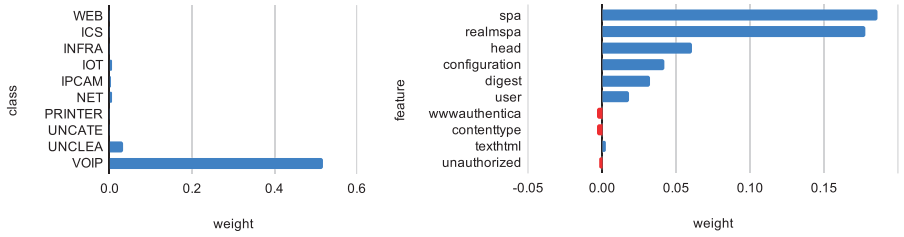
4. UNDERSTANDING THE CLASSIFICATION

Explainable classification can further increase the precision and also transfer the new knowledge back to experts. We review a randomly selected device from each class in an attempt to understand the classification and to evaluate options for improving it. We present the calculated prediction of classes and the most impactful weights of the features determining the likely classes.

A Cisco IP telephony device classified as VOIP is presented in Figure 3. While an expert would focus on the keywords “Cisco” and “SPA”, the classifier selects “spa” as the highest weight feature and disregards “cisco” as manufacturing a large variety of NET devices. While authentication headers are more indicative of other lower power

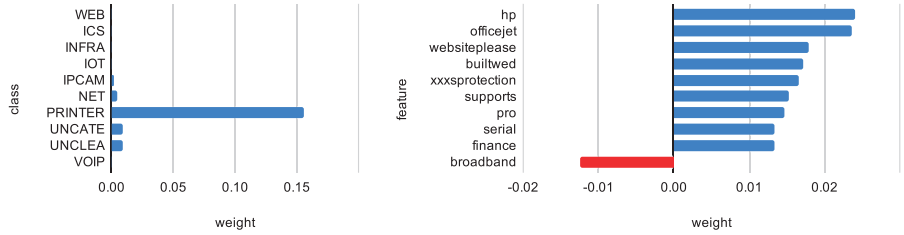
and cheaper devices and have negative weight in this case, this is counterweighted by a slightly more complex and secure variant instead of plain text.

FIGURE 3: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR A VOIP DEVICE



A PRINTER device is presented in Figure 4. The highest weight features "hp" and "officejet" correspond to one of the most common printer series covered by most static rule sets. The feature "broadband" is weighted negatively as is more expected in the context of networking devices. The "finance" keyword is part of the network name feature, which is not common in other randomly reviewed devices here.

FIGURE 4: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR A PRINTER DEVICE

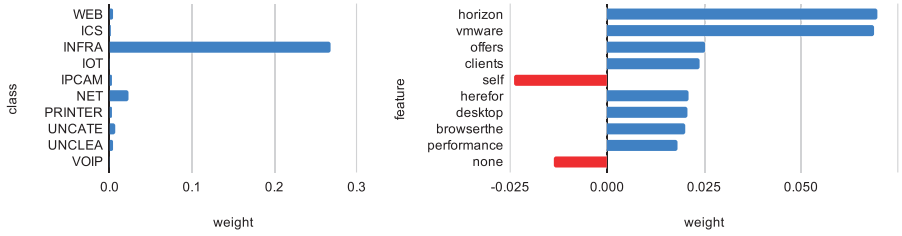


A smart home automation device from LOXONE classified as IOT is presented in Figure 5. While none of the high-weight features identifies the vendor or model, which is the way an expert would write a static rule for this device, the keyword "webinterface" for the interface and response headers has the highest weight. At the same time, security headers are weighted negatively, indicating that the model expects IOT devices to have less security features. Interestingly, a network name feature consisting of "austria" and "telekom" indicates that the manufacturer, based in Austria, has a high presence in Austrian networks. While this can be intuitively recognised by an expert, the variety of devices and complexity of the rule has prevented this from being implemented in static classification rule sets.

FIGURE 5: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN IOT DEVICE

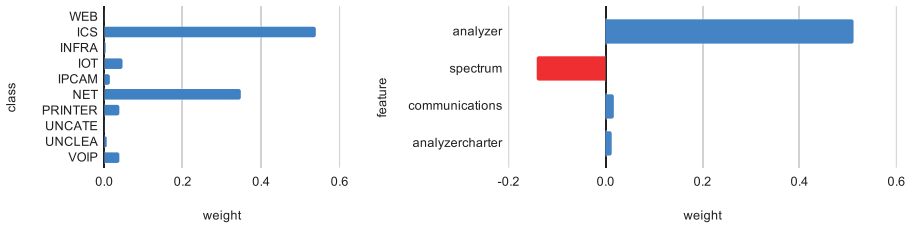
A VMware Horizon device classified as INFRA is presented in Figure 6. By the definition of the class, most VMware solutions match INFRA, thus the high weight of the keyword "vmware", as well as all other classes assigning a negative value to it, is unsurprising. The product keyword is also expected; static classification rule sets might contain a simple rule matching these two keywords together.

FIGURE 6: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN INFRA DEVICE



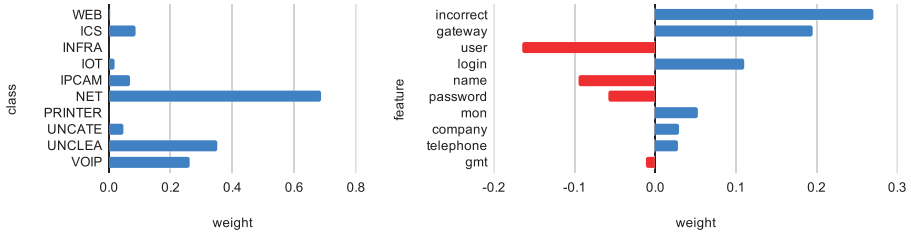
From the randomly selected devices, the spectrum analyser has the least features and is classified as ICS and presented in Figure 7. ICS devices can have the least properties of the responses that can be extracted as features. Rich response features typically weight heavily against the device being classified as an ICS. While the combination of "spectrum" and "analyser" can be evident for humans, these are treated as separate features and "spectrum" is weighted against this class while being weighted heavily in favour of some other classes. This identifies an issue with introducing network names as a feature, in this case, likely the large ISP named Spectrum, suggesting that the network name feature should be treated differently from the response features.

FIGURE 7: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN ICS DEVICE



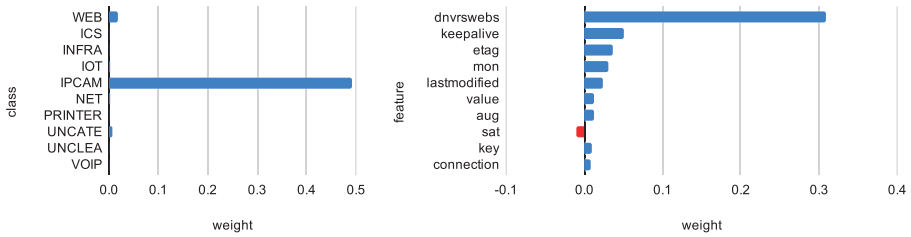
For the network device presented in Figure 8 (NET class), the second highest weight feature "gateway" is a classic keyword even in static rules. The feature set and raw response confirm that this is an unbranded residential network gateway for which even an expert is unable to extract more information without active probing. This feature is not unique to the NET class. It might correspond to gateway functionality in an application protocol sense or display configuration debugging information for any networked device. In this particular case, this feature is weighted in favour of only the VOIP class. Most of the remaining determining features consist of authentication interface keywords, including the highest weight feature "incorrect" indicating failed authentication. The way an authentication interface is presented has a high weight in determining the class.

FIGURE 8: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR A NET DEVICE



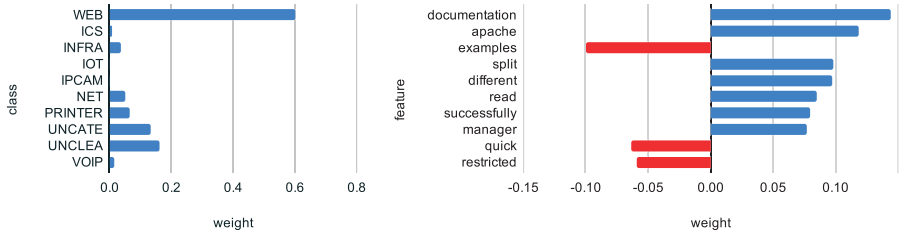
A Hikvision networked surveillance device classified as IPCAM is presented in Figure 9. The "dnvrswebs" is a software version unique to IP cameras and video recorders and thus is weighted heavily. In general, it is weighted negatively against all other classes. Most static rule sets have this as a simple match rule to reliably classify IP cameras.

FIGURE 9: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN IPCAM DEVICE



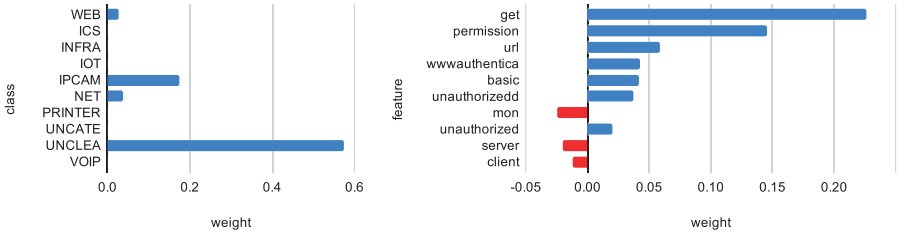
The WEB device presented in Figure 10 is an Apache Tomcat interface allowing the deployment and management of web applications. While the feature "apache" has a high weight in determining the class, it is not always the case, otherwise a blanket static rule would suffice. In general, it has a negative weight on the UNCLEAR class where no web sites are expected. The keyword "restricted", generally associated with web interface authentication, has a significant negative weight.

FIGURE 10: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR A WEB DEVICE



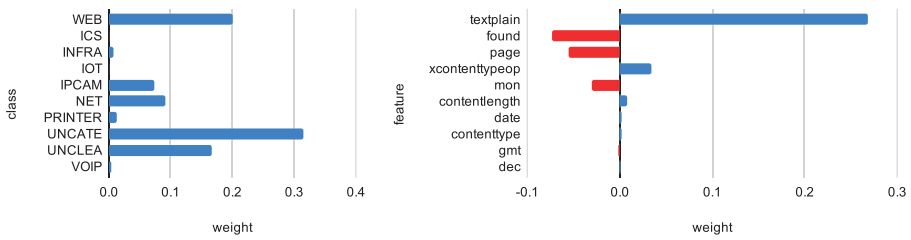
A device classified as UNCLEAR and likely an embedded device without a determinable functionality but definitely not a generic web site is presented in Figure 11. Keywords related to HTTP basic authentication and the displayed message are weighting in favour of this class; while the presence of the Server header revealing software name and version is weighting against, as it is often a high-weight feature, in this case, it is a generic embedded software having many uses.

FIGURE 11: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN UNCLEAR DEVICE



An UNCATEGORIZED device that cannot be determined as part of any class is presented in Figure 12. This device has a small feature set (all generic) but not as small as the most basic embedded devices. The generic response headers are sufficient to be also those of a website or service not handling default requests. In general, plain text content type, which is the heaviest weighted feature, corresponds to an unformatted output of mostly short error messages. From this set of features, an expert is not able to reliably determine the class either.

FIGURE 12: CLASS PREDICTIONS AND FEATURE WEIGHTS FOR AN UNCATEGORIZED DEVICE



While there can be identified cases that are common and covered by static classification rule sets even within these few random examples, a more complex classification matching expert intuition can be seen. These types of cases can be classified individually by an expert, but defining all of that into static rules is not feasible, not only because of the sheer number of rules but also the complexity which would require statistical calculations to formalise the intuition.

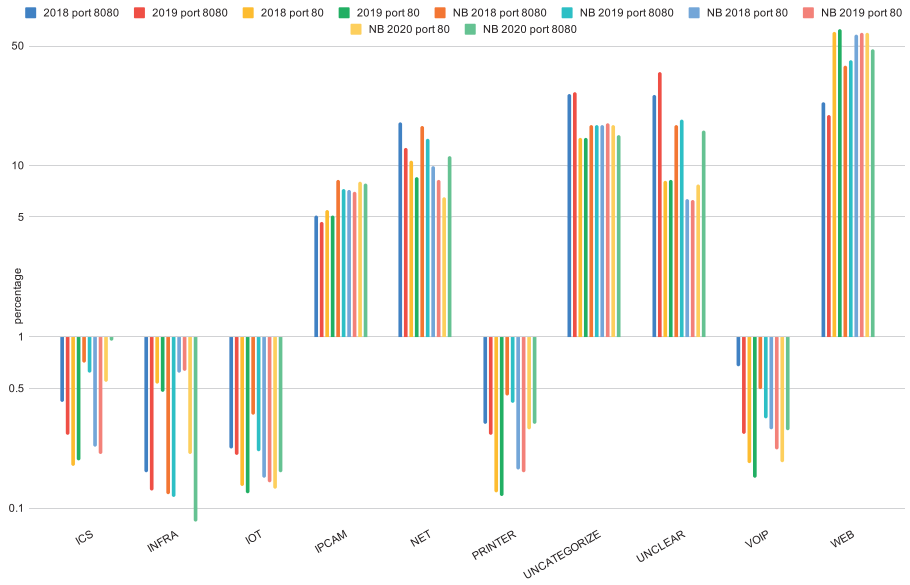
5. CLASSIFICATION RESULTS

The relative class distribution is presented in Figure 13; the Naive Bayes classifier results developed in this paper are prefixed NB. The remaining classification data are based on neural network results from [2]; the raw scan data from the same source is used to test the Naive Bayes classification for 2018 and 2019, while the 2020 data set has been created specifically for this research.

While classification differences can be easily observed, they are explained by the varying accuracy ranges between different methods. Although the goal of this research was not to analyse the classification, we can identify the main trends in Naive Bayes classification. An increase in ICS devices is unexpected in light of worldwide efforts to disconnect these devices from the Internet; most likely these are new deployments of low impact automation devices. The decrease of INFRA devices is expected with a shorter lifecycle of deployments and new deployments following better security

practices. The stable proportion of IOT devices is positive, considering the increasing number of new deployments. IP cameras, which often require remote reachability, see a slight increase, and by contrast, NET devices, which do not, see a significant decrease. IP telephony devices experience a stable decrease.

FIGURE 13: THE NEW NB CLASSIFICATION APPLIED FOR 2018–2019 AND NEURAL NETWORK CLASSIFICATION



6. CONCLUSIONS AND DISCUSSION

Device classification is an important emerging research field. While existing neural network classifiers already provide classification with high levels of accuracy [2], [9], the results are not always understandable by human experts. In some of these cases, it is hard to distinguish who is in the right. An expert often has the ability to validate his predictions through active probing and other external sources of intelligence. But what if the device is not present on the Internet anymore? This often happens because of the dynamically assigned IP address change, a device going offline or when analysing historical data sets. The expert is left with only the feature set and potentially the raw data from which features were extracted to make a decision. Features are numerous and while clues could be found and even validated using external knowledge, there is no confirmation that these were decisive features in the classification, so no correction in the classifier can be made.

Understanding the classification brings result transparency – the ability to explain the predicted class. With our suggested combined Naive Bayes and LIME approach, we were able to demonstrate a reliable method for the explainable classification of devices with a web interface being reachable on the Internet. Understanding the features used by the model for class prediction permits better analysis of the device properties and, consequently, improvements in the classification accuracy via a more targeted handling of device data and feature filtering. This approach supports a better general understanding of higher risk potentially vulnerable devices on the Internet and, subsequently, can increase not only security for the device owner but also overall security.

REFERENCES

[1] Y. Jia, B. Han, Q. Li, H. Li, and L. Sun, "Who owns Internet of Thing devices?," *International Journal of Distributed Sensor Networks*, vol. 14, no. 11, Nov. 2018, doi: 10.1177/1550147718811099.

[2] A. Lavrenovs, R. Graf, and K. Heinaaro, "Towards Classifying Devices on the Internet Using Artificial Intelligence," in *2020 12th International Conference on Cyber Conflict (CyCon)*, Estonia, May 2020, pp. 309–325, doi: 10.23919/CyCon49761.2020.9131713.

[3] A. Lavrenovs and G. Visky, "Exploring features of HTTP responses for the classification of devices on the Internet," presented at the 2019 27th Telecommunications Forum (TELFOR), Belgrade, Serbia, Nov. 2019, doi: 10.1109/TELFOR48224.2019.8971100.

[4] A. Lavrenovs and G. Visky, "Investigating HTTP response headers for the classification of devices on the Internet," presented at the 2019 IEEE 7th IEEE Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Liepaja, Latvia, Nov. 2019, doi: 10.1109/AIEEE48629.2019.8977115.

[5] A. Mirian *et al*., "An Internet-wide view of ICS devices," in *Proceedings of 2016 14th Annual Conference on Privacy, Security and Trust (PST)*, Dec. 2016, pp. 96–103, doi: 10.1109/PST.2016.7906943.

[6] M. Dahlmanns, J. Lohmöller, I. B. Fink, J. Pennekamp, K. Wehrle, and M. Henze, "Easing the Conscience with OPC UA: An Internet-Wide Study on Insecure Deployments," in *Proceedings of the ACM Internet Measurement Conference*, Virtual Event USA, Oct. 2020, pp. 101–110, doi: 10.1145/3419394.3423666.

[7] X. Feng, Q. Li, H. Wang, and L. Sun, "Acquisitional Rule-based Engine for Discovering Internet-of-Things Devices," in *27th USENIX Security Symposium (USENIX Security 18)*, Baltimore, MD, Aug. 2018, pp. 327–341, [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/feng

[8] M. Nawrocki, T. C. Schmidt, and M. Wahlisch, "Uncovering Vulnerable Industrial Control Systems from the Internet Core," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, Budapest, Hungary, Apr. 2020, pp. 1–9, doi: 10.1109/NOMS47738.2020.9110256.

[9] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of IoT devices in the cyberspace," *Comput. Netw*., vol. 148, pp. 318–327, Jan. 2019, doi: 10.1016/j.comnet.2018.11.013.

[10] Y. Meidan *et al*., "ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis," in *Proceedings of the Symposium on Applied Computing*, Marrakech Morocco, Apr. 2017, pp. 506–509, doi: 10.1145/3019612.3019878.

[11] A. Sivanathan *et al*., "Characterizing and classifying IoT traffic in smart cities and campuses," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Atlanta, GA, May 2017, pp. 559–564, doi: 10.1109/INFCOMW.2017.8116438.

[12] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Behavioral Fingerprinting of IoT Devices," in *Proceedings of the 2018 Workshop on Attacks and Solutions in Hardware Security - ASHES '18*, Toronto, Canada, 2018, pp. 41–50, doi: 10.1145/3266444.3266452.

[13] P. Yadav, A. Feraudo, B. Arief, S. F. Shahandashti, and V. G. Vassilakis, "Position paper: A systematic framework for categorising IoT device fingerprinting mechanisms," in *Proceedings of the 2nd International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things*, Virtual Event Japan, Nov. 2020, pp. 62–68, doi: 10.1145/3417313.3429384.

[14] A. Ignatiev, "Towards Trustable Explainable AI," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, Yokohama, Japan, Jul. 2020, pp. 5154–5158, doi: 10.24963/ijcai.2020/726.

[15] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, 2017, vol. 30, pp. 4765–4774, [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf
[16] N. L. Tsakiridis *et al*., "Versatile Internet of Things for Agriculture: An eXplainable AI Approach," in *Artificial Intelligence Applications and Innovations*, vol. 584, I. Maglogiannis, L. Iliadis, and E. Pimenidis, Eds. Cham: Springer International Publishing, 2020, pp. 180–191.
[17] I. Garcia-Magarino, R. Muttukrishnan, and J. Lloret, "Human-Centric AI for Trustworthy IoT Systems With Explainable Multilayer Perceptrons," *IEEE Access*, vol. 7, pp. 125562–125574, 2019, doi: 10.1109/ACCESS.2019.2937521.
[18] S. L. Y. Lam and Dik Lun Lee, "Feature reduction for neural network based text categorization," in *Proceedings. 6th International Conference on Advanced Systems for Advanced Applications*, Hsinchu, Taiwan, 1999, pp. 195–202, doi: 10.1109/DASFAA.1999.765752.
[19] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. New York: Cambridge University Press, 2008.
[20] The ZMap Team, "The ZMap Project." https://zmap.io/ (accessed Aug. 30, 2017).
[21] Md. S. Islam, S. M. Khaled, K. Farhan, Md. A. Rahman, and J. Rahman, "Modeling Spammer Behavior: Naive Bayes vs. Artificial Neural Networks," in *2009 International Conference on Information and Multimedia Technology*, Jeju Island, Korea (South), 2009, pp. 52–55, doi: 10.1109/ICIMT.2009.48.
[22] T. M. Mitchell, "Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression," in *Machine Learning*, 2nd-draft ed., 2017.
[23] X. Daniela, C. Hinde, and R. Stone, "Naive Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages," *International Journal of Computer Science Issues*, vol. 4, 2009.
[24] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA, Aug. 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.