Multi-label Classification of Hosts Observed through a Darknet

Enzo d'Andréa, Jérôme François, Olivier Festor, Mehdi Zakroum

Multi-label Classification of Hosts Observed through a Darknet

Enzo d'Andréa
Jérôme François
INRIA Nancy Grand EST - LORIA
Nancy, France
enzo.d-andrea@inria.fr
jerome.francois@inria.fr

Olivier Festor
Université de Lorraine - LORIA
Nancy, France
olivier.festor@inria.fr

Mehdi Zakroum
Université de Lorraine - LORIA
Nancy, France
TICLab, Int. University of Rabat
Rabat, Morocco
mehdi.zakroum@loria.fr

***Abstract*—To observe compromised hosts at Internet-scale, a darknet or network telescope collects Internet background radiation that includes large-scale phenomena like DDoS (Distributed Denial-of-Service) or scanning. Gathered data is however very partial and labeling such traffic to precise activities thanks to external databases is far from being satisfactory (8.4% of IP addresses in our case). In addition, as compromised hosts are used for multiple malicious activities, they cannot be classified in a unique category. We propose in this paper a new multi-label classification method by representing traffic generated by a host as a graph and leveraging machine learning algorithms (Node embedding and Graph Convolutional Networks). From partial information about IP addresses, our method can label addresses with a precision of 0.80 and recall of 0.81.**

I. Introduction

Darknets or network telescopes have been widely used to observe cyber-threats at the Internet scale like botnet [1] and scans [2]. A darknet silently collects all incoming traffic, without creating outgoing traffic. Hence neither replies nor real connections to services can be monitored. This traffic does not have an inherent signification and external sources such as AbuseIPDB or NERD would be required to give a signification to the data. The characterization of this partial traffic and the involved hosts is thus a challenging task. Moreover, a host can be involved in multiple types of malicious activities like DDoS, and regular scans. Hence, this is a multi-label classification problem, unlike binary or multi-class classification tasks which have been widely considered.

Our goal is to classify each individual host (IP address) observed in a darknet. Our approach relies on graph representations of hosts behaviors. Although graph-based models for network traffic have been already proposed [3], [4], they assume the ability to capture active interactions. In contrast, due to the absence of active connections in a darknet, we design a model able to embed the whole behavior of a host based on its probing activity (connection attempts).

To predict the labels of the graphs, several types of neural networks are used: an auto-encoder creates TCP/UDP port embeddings integrated into the graph nodes, and Graph Convolutional Networks (GCNs [5]) to infer latent representations of the graphs. The method has been applied on one month of traffic captured by a /20 darknet, for which less than 10% of the IP addresses were labeled thanks to an external database.

The rest of this paper is organized as follows: Section II presents related works, Section III formally defines the problem. Section IV describes the proposed method. Evaluation is detailed in Section V. Section VI concludes the paper.

II. Related works

Darknet traffic can be analyzed to serve multiple purposes as presented in [6]. In [2], the objective is to characterize scanning campaigns. Tracking botnets with the help of a darknet has been also widely investigated [7], [8]. Soro et al. [9] classify Autonomous Systems (ASs) using a unsupervised machine learning method. TCP traffic is assembled into a bipartite graph, from which ASs are grouped thanks to a community detection algorithm. This method however does not consider UDP and ICMP traffic. Unlike these works, our goal is to label each individual host observed in a darknet assuming several types of malicious activities.

Shaikh et al. [10] leverage supervised machine learning algorithms to classify traffic as one of [*Scanning*, *Backscatter*, *Misconfiguration* or *Other*] for IoT devices observed through a darknet. Our work is similar by nature but differs from the set of activities to be identified (7 in total) and by representing the traffic of a host as a graph. In [11], the authors analyze the sequence of ports targeted by the different IP addresses to group them when they share common patterns.

Our graph-based model is close to other works representing network traffic as graphs before using graph-based machine learning techniques[12], [13]. Sun et al. [12] represent encrypted flows as nodes in a graph while GCNs are then used to classify flows. Pan et al. [13] also rely on GCNs but for anomaly detection (binary problem). In this paper, the network traffic generated by each host is represented as a single graph to embed its whole behavior and a GCN is used to associate it to multiple labels representing its activities.

III. Problem definition

A. Background on darknets

A darknet [14], also known as a network telescope or a sinkhole, allows us to passively monitor internet-wide ac-

tivities. It consists of a subnet of unused but reachable IP addresses. Because such IP addresses have never been in use, they represent hosts that have never communicated with any other host, and are therefore not expected to receive any legitimate traffic. Such unused addresses receive a significant amount of unsolicited traffic from internet-wide scans [15], side effect of DDoS attacks [16] or misconfigurations [17].

Because of its passive nature, the observation of attacker activities is limited. Therefore, we can only observe the initiating packets, mostly with no payload.

B. Problem statement and Objective

Identifying the type of activities performed by each observed host in a darknet could help in tracking ongoing Internet-wide scale attacks and thus better defend against them [18]. Inferring such information without prior knowledge is known to be difficult [19]. Often, external services or databases like AbuseIPDB[1] or NERD[2] are leveraged. They are community-based tools whose aim is to collect reports from users having experienced attacks in their own networks. However, the coverage of this source of data is very limited. In our case, as shown in Section V-B, only 8.4% of the monitored IP addresses in the darknet can be labeled using AbuseIPDB.

To tackle this issue, our method automatically labels IP addresses observed by a darknet according to their probing activities. We refer here to a multi-labeling rather than a multi-class classification problem as a host can be involved in multiple activities. On one hand, the main challenge is the partial nature of data because only the first packets of connection attempts are observed. On the other hand, specific probing or scanning strategies can convey information about reconnaissance phase of an attack [20], and so possibly the type of activities the host is involved in.

Formally, assuming a set of hosts $H = \{h_1, h_2, \ldots, h_n\}$, each represented by its IP address, we define the flow of packets originating from h_i as $f^{(i)} = \langle p_0^{(i)}, p_1^{(i)}, ..., p_m^{(i)} \rangle$. $p_k^{(i)}$ represents the k^{th} packet received from the host h_i and is defined as a tuple $p_k^{(i)} = \langle timestamp, protocol, source\ port, destination\ port, TCP\ flags, ICMP\ type, ICMP\ code \rangle$. Obviously, some of these packet features are exclusive, like TCP flags and ICMP code. Each h_i is labeled with a set of labels $L_i \in L = \{l_1, l_2, \ldots, l_k\}$, extracted form a knowledge base combining reports from different sources. Our goal is to define a function $\mathcal{L}$ that maps $H_{\text{def}} \subset H$ to L and that predicts the set of labels L_u to which a source host $h_u \in H$, $h_u \notin H_{\text{def}}$ belongs. $\mathcal{L}$ is partially defined for a subset of hosts H_{def}.

IV. Method

A. Approach Overview

We propose a supervised classification technique taking as inputs the traffic from the observed hosts. The main difficulty resides in obtaining a meaningful representation from the latter. In our case, we derive a graph representing how a host acts towards the different network services targeted. Figure 1 depicts an overview of our approach with 4 main steps:

1) *Preprocessing* to exclude noise or irrelevant data.
2) *Ports embeddings* to provide relevant features for representing TCP and UDP destination port numbers in Host graphs.
3) *Host graphs extraction* to represent each source IP address as a behavioral graph.
4) *Host classification* leveraging GCNs to infer the labels from the extracted graphs. As it is a supervised technique, *AbuseIPDP* is used as an external source of information to label training data.

B. Network Flows Preprocessing

To discard meaningless data, three different filters are applied. The first one removes isolated packets, i.e. packets sent by hosts that have sent at most 1 packet in 7 rolling days. They are considered to have not enough relationship with other packets to infer a specific behavior.

The second filter discards the vertical scans. Such scans are easily detected (when a host probes many port numbers), and their objectives are well defined. They are thus excluded from this study. The boundary between vertical and non-vertical scans is empirically defined in Section V-A

The third filter is called the periodic pattern filter. Its objective is to discard the hosts scanning ports in a fixed predefined manner, e.g. {80, 443, 80, 443,...} or {2, 4, 6,...}. Indeed, they represent a deterministic behavior that does not require complex analysis, again out of scope of our objective.

C. Port embeddings

Although TCP/UDP ports are commonly representative of network services, their numeric values do not encode a meaning, so using their raw format as inputs to a learning task is inefficient. To tackle this issue, we infer latent representations of ports by encoding their semantics based on how often a pair of ports appear in a probing sequence. We rely on the graph representation proposed in [20], but with both TCP and UDP ports to capture inter-port relationship when they are scanned by attackers. In details, a directed weighted graph $G_p = (N_p, E_p, \omega_p)$ is created with:

- N_p The set of nodes of the graph. Each node represents a unique port, either TCP or UDP.
- E_p The set of edges of the graph. An edge $e_{i,j}$ from port p_i to p_j exists if port p_j has been probed right after p_i from the same source IP address.
- ω_p The weight function for edges defined as $\omega_p(e_{i,j})$; the weights calculated following the method in [20].

The weights are derived from the distribution θ of all $\theta_{i,j}$, the number of times port p_j follows p_i in all network flows, and its quartiles Q_1 and Q_3:

$$\omega'_{i,j} = \frac{\theta_{i,j} - Q_1(\theta)}{Q_3(\theta) - Q_1(\theta)}$$

[1]https://www.abuseipdb.com/
[2]https://nerd.cesnet.cz/

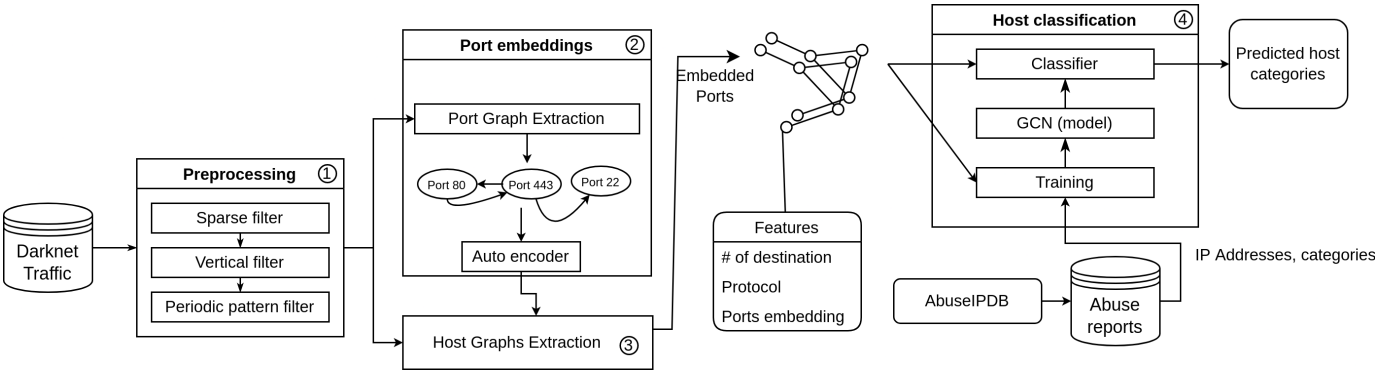


Fig. 1: Overview

Because values below Q_1 become negative, final weights are shifted using the minimum and maximum values:

$$\omega_{i,j} = (max_{i,j}(\omega'_{i,j}) - min_{i,j}(\omega'_{i,j})) - (\omega'_{i,j} - min_{i,j}(\omega'_{i,j}))$$

Unlike the authors of [20] considering the shortest path distances between pairs of nodes to infer a similarity, we create an initial embedding for each port consisting of the distances between the port and a set of randomly selected ports. We then train an auto-encoder from this initial embedding. Each layer consists of a fully connected layer with batch normalization and a Rectified Linear Unit (ReLU) activation. The encoder produces a latent representation for the network ports through a linear hidden layer. Further details are provided in Section V-C.

D. Host graphs extraction

To create a representation of the behavior of host i, packets $p_k^{(i)}$ of the flow $f^{(i)}$ are grouped according to the tuple $\langle Protocol,\ Destination\ Port,\ TCP\ Flags,\ ICMP\ Type/Code\rangle$ to form a unique node in the graph $G_h = (N_h, E_h, \omega_h)$:

N_h The set of nodes. Each node represents a group of packets as defined above with the following features:
- Embedding of destination IP addresses (3 values): number of addresses targeted in the darknet; A value equal to 1, -1 or 0 if the addresses are reached in increasing, decreasing or arbitrary order; A binary value equal to 1 if all consecutive addresses between the minimal and maximal addresses are reached,
- Source ports embedding: number of ports, one-hot encoding of the class of the ports: well-known, registered or dynamic according to [21].
- Embedding of the destination port (see Section IV-C),
- One hot encoding of the protocol (TCP/UDP/ICMP),
- TCP Flags on 8 bits, -1 for UDP and ICMP packets,
- ICMP Type and Code, -1 for TCP and UDP packets

E_h The set of edges. An edge $e_{i,j}$ from node n_i to n_j exists if a packet corresponding to n_j is followed by a packet corresponding to n_i or vice-versa.

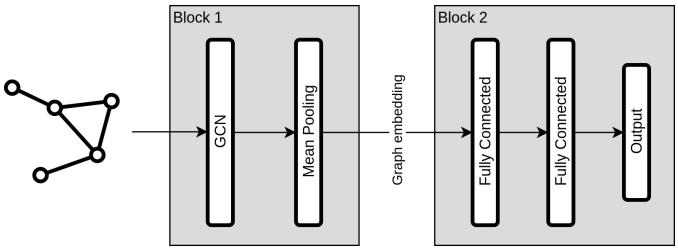


Fig. 2: Host classification model

ω_h The weight function for edges defined as $\omega_h(e_{i,j})$ the number of occurrences of a packet from n_j followed by a packet from n_i or vice-versa.

E. Host multi-labeling

Hosts are classified using a two-block model but trained as a whole during the learning process. As shown in Figure 2, the first block consists of a GCN layer [5] transforming the graph structure and node features into node embeddings. The layer-wise propagation rule of a GCN layer is as follow:

$$H^{(l+1)} = f(H^{(l)}, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}),$$

where $H^{(l)}$ is the representation of nodes features on the l^{th} layer, $H^{(0)}$ being the initial features as defined in Section IV-D, $\hat{A} = A + I$, where I is the identity matrix and A the adjacency matrix of the graph being classified, and $\hat{D}$ the diagonal node degree matrix of $\hat{A}$. Following the convolutional layers, a mean pooling layer is added to aggregate the inferred node embeddings into an entire graph embedding.

The second block is a classifier composed of fully connected Perceptron layers with batch normalization and ReLU activation. The output layer consists of a single fully connected Perceptron layer with a sigmoid activation function. From the embedding of a graph (from the first block), the outputs are the labels that correspond to the host behavior. Each label is associated to an output node of the model. During the inference stage, if the value in an output node is higher than a threshold τ (0.5 by default), the corresponding label is predicted as an output.

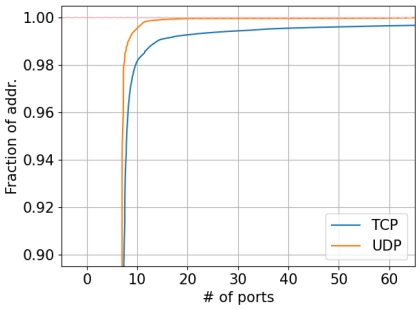


Fig. 3: Weekly number of ports reached per address (cumulative)

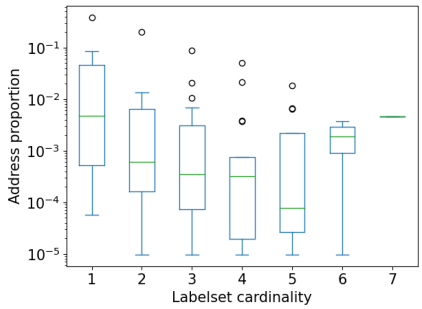


Fig. 4: Label set size per IP address distribution

TABLE I: Labels distribution (Top 7 categories)

Category	ID	% of addr.
Port Scan	14	87.34%
Hacking	15	54.79%
Brute-Force	18	26.84%
Web App Attack	21	11.65%
SSH	22	11.00%
Exploited Host	20	3.08%
Bad Web Bot	19	2.31%

V. Results and Evaluation

A. Darknet Dataset

Our dataset consists of packets collected by a /20 network telescope over March 2021 from 2 millions unique source IP addresses. The number of packets per IP addresses is extremely unbalanced, with quartiles of 1 (Q1) and 9 (Q3) packets but a maximum of 27 millions and a mean of 340.

Roughly 20 millions TCP packets are received each day, 1.5 million for UDP and 100,000 for ICMP. The weekly number of destination ports reached per source IP address is shown in Figure 3. 99% of the source hosts reach less than 20 TCP ports per week and 99.9% reach less than 15 UDP ports. These values are used as the limit between vertical and non-vertical scans (Section IV-B). After preprocessing, 1.2 million IP addresses are kept with on average 163 packets per IP address and a total number of 59,385 unique TCP ports and 24,091 unique UDP ports.

B. Host Labeling

As stated in Section III-B, $\mathcal{L}$ is partially defined thanks to *AbuseIPDB* and represents the ground truth used in our evaluation. This database covers 24 distinct categories including *Port Scan*, *DDoS* and *SSH attacks*. The reports of March 2021 cover a total of 289,787 IP addresses of the darknet. Among these addresses, 103,723 (8.4% of all addresses from the darknet) are considered labeled because they have at least two reports in one of the 7 most represented categories. Description of each category can be found at *AbuseIPDB*[3]. All other categories are excluded since they are present in less than 2% of the labeled IP addresses. Table I details the overall proportion of each individual label.

The distribution of the number of labels per IP address (labelset size) is depicted in Figure 4. Around 50% of IP addresses have a single label.

C. Hyperparameters

Hyperparameters (HP) for the auto-encoder are the number of ports chosen as references, the size of the latent representation and the number of hidden layers in each halves of the auto-encoder. For the classifier, the parameters are the size of the first layer, and the number of layers in each part of the model (GCN and Fully Connected). Layer sizes decreases exponentially from the initial layer size to the final size, either the size of the latent representation or the number of categories to predict. They have been tuned with a grid-search and results are summarized Table II.

TABLE II: HP tuning (bold values are selected)

Auto-encoder			Classifier		
# of ref.	**Emb. size**	**# layers**	**Layer size**	**# GCN**	**# FC**
512	**8**/16/32	**0**/1/2	64/**128**/256	1/**2**/3	1/**2**/3
1024	**8**/16/32	**0**/1/2			
2048	**8**/16/32	**0**/1/2			

Empirically, auto-encoder and classifier are trained with Adam optimizer, learning rate of 0.001 (respectively 0.005), a 256 (32) batch size and over 50 (20) epochs, stopped when no improvement are made on the training loss after 5 (3) consecutive epochs.

The models are trained on a machine with a 40-core *Intel Xeon* CPU and 128 GB of RAM. The whole preprocessing task needs roughly one hour to be completed, as for the creation of port embeddings and the extraction of host graphs. Training the classification model needs 30 to 40 minutes depending on the hyperparameters being tested.

D. Overall per-label performance

In the first experiment, we assess the ability of our model to predict each individual category. In multi-label problems, prediction metrics (precision, recall, F1 score) are computed

TABLE III: Per-label prediction accuracy

Category	Pre	Rec	F1	Support
Port Scan	0.90	0.98	0.94	87.1%
Hacking	0.67	0.73	0.70	54.6%
Brute Force	0.68	0.50	0.57	26.7%
Web App Attack	0.78	0.47	0.51	11.4%
SSH	0.56	0.43	0.55	10.4%
Exploited Host	0.40	0.54	0.46	3.0%
Bad Web Bot	0.25	0.58	0.35	2.2%

[3]https://www.abuseipdb.com/categories, accessed on 14 Sep 2022

for each sample by comparing true and predicted labels (categories) and then averaged over all the samples. Such metrics are qualified as sample- or example- based [22]. The dataset is divided into 5 random parts, one for testing and the others for cross-validation. Training data is balanced by oversampling with ML-ROS [23] until a 25% increase in size.

The sample precision on the test data is 0.80, with a recall of 0.81 and a F1-score of 0.76, assuming a threshold $\tau = 0.5$. However, the accuracy highly varies depending on the category as summarized in Table III. Per label F1-scores vary from 0.94 to 0.35 and are mostly correlated with the proportion of samples of the given label. For instance, the most present category *Port Scan* is the most predictable one, while *Bad Web Bot* and *Exploited Host* have rather mixed results.

E. Impact of the threshold

Because our goal is to improve the characterization of the hosts in a darknet, labeling all IP addresses is not necessary but it must be precise for those which are labeled, even if it decreases the recall.

The trade-off between precision and recall is adjusted thanks to the threshold τ between a positive or negative prediction. On one hand, when τ is higher, only labels with higher output are considered and so the precision is improved as shown in Figure 5(a). Category 14 (*Port Scan*) depicts a very high precision throughout the whole range. This can be explained because it is present in the majority of the test samples (87.1%). Category 19 *Bad Web Bot* however, depicts a significantly lower precision. It is also the least represented category.

On the other hand, the recall value consistently decreases with the increase of τ (Figure 5(b)). Category *Port Scan* is again the category with the highest score while the categories *SSH* and *Bad Web Bot* have the lowest score.

Based on the F1-score (Figure 5(c)), it can be noted that while most of the categories reach their maximum value with $\tau \sim 0.4$, the least represented one, *Bad Web Bot*, reach its peak with $\tau \sim 0.6$.

F. Per-host performance

In order to evaluate the ability of our model to predict the labels of a host, we consider the following metrics from [24]:

- Coverage error: the number of top predictions in the output layer needed to predict all the true labels. In average, our method needs to predict 2.24 labels while the average number of true labels per IP address is 1.96. This small difference indicates few wrong predictions before finding all labels.
- Label ranking loss: the number of times the output value of a wrong prediction is higher than a right one. The value is rather low (0.039), a few wrong labels are usually assigned along with the right ones to a host.
- Subset accuracy: the strictest metric measures whether or not all the exact labels are predicted. Due to at least one wrong prediction, it is low (0.405).

VI. Conclusion

This paper introduced a graph model to represent the behavior of the hosts in a darknet. The sparse nature of traffic flows forces to analyze the behavior of a host as a whole rather than a per flow or per connection. Hence, the behavior is a composed of multiple activities or categories leading so to a multi-label classification problem. We leveraged an auto-encoder to derive machine learning-usable representations of TCP/UDP port numbers, a GCN to embed node and structural graph information for multi-labeling purposes. We analyzed data collected over one month on a /20 darknet, with a total of 1.2 million hosts. The results showed an acceptable performance even if all exact labels of an IP address cannot be predicted.

While the port embedding is currently derived from the measured distance from each port to randomly chosen ports, future work will improve this selection.

References

[1] E. Malécot and D. Inoue, "The carna botnet through the lens of a network telescope," in *6th International Symposium on Foundations and Practice of Security*. Springer-Verlag, 2013.

[2] P. Richter and A. Berger, "Scanning the scanners: Sensing the internet from a massively distributed network telescope," in *Internet Measurement Conference*. ACM, 2019.

[3] M. Iliofotou, H.-c. Kim, M. Faloutsos, M. Mitzenmacher, P. Pappu, and G. Varghese, "Graph-based p2p traffic classification at the internet backbone," in *IEEE INFOCOM Workshops*, 2009.

[4] A. A. Daya, M. A. Salahuddin, N. Limam, and R. Boutaba, "Botchase: Graph-based bot detection using machine learning," vol. 17, no. 1, 2020, pp. 15–29.

[5] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016.

[6] M. Bailey, E. Cooke, F. Jahanian, A. Myrick, and S. Sinha, "Practical darknet measurement," in *40th Annual Conference on Information Sciences and Systems*, 2006.

[7] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescapé, "Analysis of a /0 stealth scan from a botnet," *IEEE/ACM Transactions on Networking*, 2015.

[8] T. Ban, L. Zhu, J. Shimamura, S. Pang, D. Inoue, and K. Nakao, "Detection of botnet activities through the lens of a large-scale darknet," in *Neural Information Processing*. Springer, 2017.

[9] F. Soro, M. Allegretta, M. Mellia, I. Drago, and L. M. Bertholdo, "Sensing the noise: Uncovering communities in darknet traffic," in *Mediterranean Communication and Computer Networking Conference*, 2020.

[10] F. Shaikh, E. Bou-Harb, J. Crichigno, and N. Ghani, "A machine learning model for classifying unsolicited iot devices by observing network telescopes," in *14th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2018.

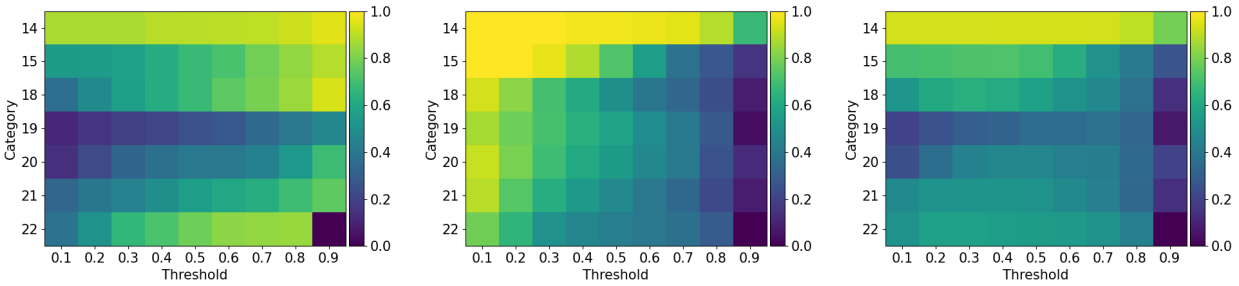
[11] D. Cohen, Y. Mirsky, M. Kamp, T. Martin, Y. Elovici, R. Puzis, and A. Shabtai, "Dante: A framework for mining and monitoring darknet traffic," in *European Symposium on Research in Computer Security, ESORICS*. Springer, 2020.

[12] B. Sun, W. Yang, M. Yan, D. Wu, Y. Zhu, and Z. Bai, "An encrypted traffic classification method combining graph convolutional network and autoencoder," in *IEEE 39th International Performance Computing and Communications Conference*, 2020.

[13] Y. Pan, X. Zhang, H. Jiang, and C. Li, "A network traffic classification method based on graph convolution and lstm," 2021.

[14] P. S. Joshi and H. Dinesha, "Survey on identification of malicious activities by monitoring darknet access," in *Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020.

[15] A. Dainotti, A. King, and K. Claffy, "Analysis of internet-wide probing using darknets," in *ACM Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, 2012.



(a) Precision (b) Recall (c) F1 score

Fig. 5: Impact of the threshold value

[16] E. Balkanli and A. N. Zincir-Heywood, "On the analysis of backscatter traffic," in *39th Annual IEEE Conference on Local Computer Networks Workshops*, 2014.

[17] C. Fachkha, E. Bou-Harb, A. Keliris, N. D. Memon, and M. Ahamad, "Internet-scale probing of cps: Inference, characterization and orchestration analysis," in *Usenix NDSS*, 2017.

[18] S.-s. Choi, J. Song, S. Kim, and S. Kim, "A model of analyzing cyber threats trend and tracing potential attackers based on darknet traffic," *Security and Communication Networks*, 2014.

[19] J. Liu and K. Fukuda, "Towards a taxonomy of darknet traffic," in *International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2014.

[20] L. Evrard, J. Fran¸cois, and J.-N. Colin, "Attacker behavior-based metric for security monitoring applied to darknet analysis," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019.

[21] M. Cotton, L. Eggert, D. J. D. Touch, M. Westerlund, and S. Cheshire, "Iana procedures for the management of the service name and transport protocol port number registry," RFC 6335, Aug. 2011.

[22] A. F. Giraldo-Forero, J. A. Jaramillo-Garzón, and C. G. Castellanos-Domínguez, "Evaluation of example-based measures for multi-label classification performance," in *Bioinformatics and Biomedical Engineering*. Springer, 2015.

[23] F. Charte, A. Rivera, M. J. del Jesus, and F. Herrera, "A first approach to deal with imbalance in multi-label datasets," in *Hybrid Artificial Intelligent Systems*. Springer, 2013.

[24] A. N. Tarekegn, M. Giacobini, and K. Michalak, "A review of methods for imbalanced multi-label classification," *Pattern Recognition*, 2021.