Deep in the Dark

2019 IEEE International Conference on Big Data (Big Data)

Deep in the Dark: A Novel Threat Detection System using Darknet Traffic

Sanjay Kumar[1], Harald Vranken[2], Joost van Dijk[3], and Timo Hamalainen[1]

[1]Faculty of Information Technology, University of Jyvaskyla, Finland
[2]Open University of the Netherlands & Radboud University, The Netherlands
[3]SURFnet, The Netherlands

***Abstract*—This paper proposes a threat detection system based on Machine Learning classifiers that are trained using darknet traffic. Traffic destined to Darknet is either malicious or by misconfiguration. Darknet traffic contains traces of several threats such as DDoS attacks, botnets, spoofing, probes and scanning attacks. We analyse darknet traffic by extracting network traffic features from it that help in finding patterns of these advanced threats. We collected the darknet traffic from the network sensors deployed at SURFnet and extracted several network-based features. In this study, we proposed a framework that uses supervised machine learning and a concept drift detector. Our experimental results show that our classifiers can easily distinguish between benign and malign traffic and are able to detect known and unknown threats effectively with an accuracy above 99%.**

***Index Terms*—Darknet traffic, DDoS, Machine Learning, Threat Detection, Network Telescope**

I. Introduction

As the use of Internet of Things (IoT) and mobile devices is increasing rapidly, also potential threats are increasing. According to [1], there will be around 38 billion IoT connected devices by 2020, which will be used for personal and industrial purposes. The 2016 Dyn attack [2] was the largest DDoS attack ever seen and disrupted the internet for several hours. This DDoS attack originated from IoT devices that lacked appropriate security measures. Also, botnets like Hijame [3] and Reaper [4] use advanced techniques to avoid detection by traditional detection systems. These kind of threats are not limited to IoT devices and affect the entire internet eco-system.

To detect and mitigate these threats, we need novel systems that can extract attack patterns from internet traffic streams and adapt to changing behavior of adversaries in an automated way. A promising approach, adopted by many researchers, is to apply Machine Learning (ML) using supervised learning to train a threat detection system using malicious and normal traffic [5]. However, the labeling of data is very expensive, and the changing behavior of adversaries makes it challenging to cope with the issue of concept drift [5], [6].

In this paper we present a novel approach that uses darknet traffic to train a ML classifier. A darknet is a portion of routed, allocated IP space in which seemingly no active services or servers reside. However, it may contain invisible systems that do receive messages but not respond to anything, or that are part of an overlay network that can be accessed by non-standard communication protocols and ports. Traffic destined to darknets is suspicious and may be either malicious or due to misconfiguration. Monitoring systems can be setup in darknets to attract or trap attackers for intelligence gathering. Also, malware and botnets are often not intelligent enough and propagate traffic widely on the internet, also addressing darknets.

In this study, we have gathered traffic from darknet sensors and extracted features from the collected darknet traffic to find patterns that detect several threats. We trained ML classifiers using these features with darknet traffic and normal traffic generated by several benign applications. This effectively builds a framework for a threat detection system, that is based on network features extracted from darknet traffic. This threat detection system can detect known and unknown attacks of similar kind, and is also able to cope with concept drift caused by the changing behavior of adversaries.

The structure of this paper is as follows. Section 2 describes the background of the darknet ecosystem and the motivation of this study. In Section 3, we discuss previous work on darknet monitoring and their limitations. Section 3 describes our methodology, proposed ML model/framework, and the development steps. In Section 4 we present the experiments conducted in this study, and interpretation of the results obtained from the ML classifiers. Finally, in Section 5, the conclusion and future work of this research are outlined.

II. Background

This section defines various terms related to darknets and provides examples of the operation and taxonomy of darknets.

Darknet is referred to by many terms such as darkspace, unallocated reachable IP addresses, network telescope, etc. It is a 'dark' portion of IP space where seemingly no active services reside [7]. Darknet monitoring solutions can include a server which acts as a sensor to gather the packets that enter a darknet for cyber-intelligence purposes. As there is no known service running on these networks, any traffic reaching these networks is either by misconfiguration or due to malware. Most of the darknet traffic falls in the second case, including traffic from Internet scanning, malware propagation, or back-scatter DDoS events [8]. Traffic that reaches a darknet can

be used to find patterns for several purposes such as IDS, packet sniffing detection tools, DDoS detection or backscatter traffic detection. In this study, darknet traffic is gathered and features are extracted in order to detect the above-mentioned threats using ML algorithms.

III. Related Work

Nowadays, Artificial Intelligence (AI) and ML based technologies are becoming a need for threat detection systems, as the attackers also adapted the use of AI as an offensive tool. Several researchers have been working on darknet monitoring to detect threats like DDoS, large scale internet probing, and botnets. The feature sets used in their studies is small and limited to identify a certain type of attack.

In 2018, Bou-harb et al. [9] built a CSC-Detector which used real-time darknet data to detect large scale probings. The author claimed that the study is different from others, as they focused on detecting the fingerprinting activities and techniques rather than finding the fingerprinting source. They used 250GB of real darknet data for their experiments. Zhang et al. [10] proposed UnitecDEAMP in 2017, to profile malicious events in darknet traffic. They segmented and categorized the extracted flows from darknet traffic according to behavioral assessment, and were able to detect most significant malicious events. In 2017, Skrjanc et al. [11] proposed large-scale cyberattack monitoring using Cauchy possibilistic clustering. They extracted 17 traffic features from darknet packets. They obtained a detection rate of 98% for DDoS backscatter and 72.8% for non-DDoS backscatter communication using support vector machines. Balkanli et al. [12] built a classifier based on decision trees to detect back-scatter DDoS events. The CAIDA dataset was used for training purposes, and 8 features were proposed out of 21 features using chi-square and symmetrical uncertainty. Ali et al. [13] used a neural network to detect DDoS attacks using 21 features from darknet traffic, which were collected by NICT Japan. Furutani et al. [14] used 11 features of port/IP information and trained a SVM based classifier to detect DDoS backscatter communication. Their classifier obtained 90% accuracy.

IV. Methodology

We propose a framework for threat detection using darknet traffic as illustrated in Figure 1. The framework is based on the network traffic features extracted from darknet traffic and used to train ML classifiers. The framework consists of many modules including traffic generation and collection, feature extraction, processing of the dataset and building the classifiers. The framework also has a concept drift detector, which compares the performance with a threshold and measures the magnitude of drift. Also, it checks for features that produce the drift.

A. Data feed collection

Our darknet feeds were collected by SURFnet, an association that offers high-quality network services to all Dutch educational and research institutions. The feeds were gathered by a darknet sensor that monitors the traffic destined to their non-allocated address space. The address space used to collect the feeds is not published in order to protect it from adversarial poisoning of the feeds.

Our normal traffic was generated from various well-known applications while performing several tasks, such as browsing, chatting, streaming, downloading and VoIP. We used several well-known applications for that purpose, such as Facebook, Chrome, Twitter, Skype, Outlook and Youtube etc.

TABLE (I) Feature List

S. No	Feature	S. No	Feature
1	Duration	39	PacketLenMax
2	FwdPacket	40	PacketLenMean
3	BackPacket	41	PacketLenStd
4	LenFwdPacket	42	PacketLenVariance
5	LenBackPacket	43	FlowagFinCount
6	FwdPacketMin	44	FlowagSynCount
7	BackPacketMin	45	FlowagRstCount
8	FwdPacketLenMean	46	FlowagPushCount
9	FwdPacketLenStd	47	FlowagAckCount
10	BackPacketLenMin	48	FlowagUrgCount
11	BackPacketLenMax	49	FlowagCWECount
12	BackPacketLenMean	50	RatioDownUp
13	BackPacketLenStd	51	AvgPacketSize
14	FlowByte	52	AvgFwdSegSize
15	FlowPacket	53	AvgBackSegSize
16	Flow2PacketMean	54	FwdHeadPacketet
17	Flow2PacketStd	55	FwdAvgByte
18	Flow2PacketMax	56	FwdAvgPacket
19	Flow2PacketMin	57	FwdAVGBulkRate
20	Fwd2PacketMin	58	BackAvgBytes/Bulk
21	Fwd2PacketMax	59	BackAvgPacketet/Bulk
22	Fwd2PacketMean	60	BackAvgBulk Rate
23	Fwd2PacketStd	61	SubFlowowFwdPacket
24	Fwd2PacketTotal	62	SubFlowowFwdBytes
25	Back2PacketMin	63	SubFlowowBackPacket
26	Back2PacketMax	64	SubFlowowBackBytes
27	Back2PacketMean	65	InitWinbytesforward
28	Back2PacketStd	66	InitWinbytesbackward
29	Back2PacketTotal	67	Actdatapktforward
30	FwdPushFlowag	68	MinSegSizeForward
31	BackPushFlowag	69	ActiveMin
32	FwdUrgFlowag	70	ActiveMean
33	BackUrgFlowag	71	ActiveMax
34	FwdHeadByte	72	ActiveStd
35	BackHeadByte	73	IdleMin
36	FwdPacket	74	IdleMean
37	BackPacket	75	IdleMax
38	PacketLenMin	76	IdleStd

B. Feature Extraction

After collecting the darknet traffic, we extracted features from the PCAP files. Proper feature extraction is necessary to build a reliable threat detection system and to cope with concept drift. We extracted 76 features from the network traffic as listed in Table I. IP addresses and port addresses were not part of the feature list. Part of the features are extracted by the method mentioned in our prior paper [5] and also several other tools were used to further enhance the feature set [15]–[17].

C. Prepossessing and Model Training

We used the Microsoft Azure ML platform [18] for preprocessing of the dataset and training of the ML model.

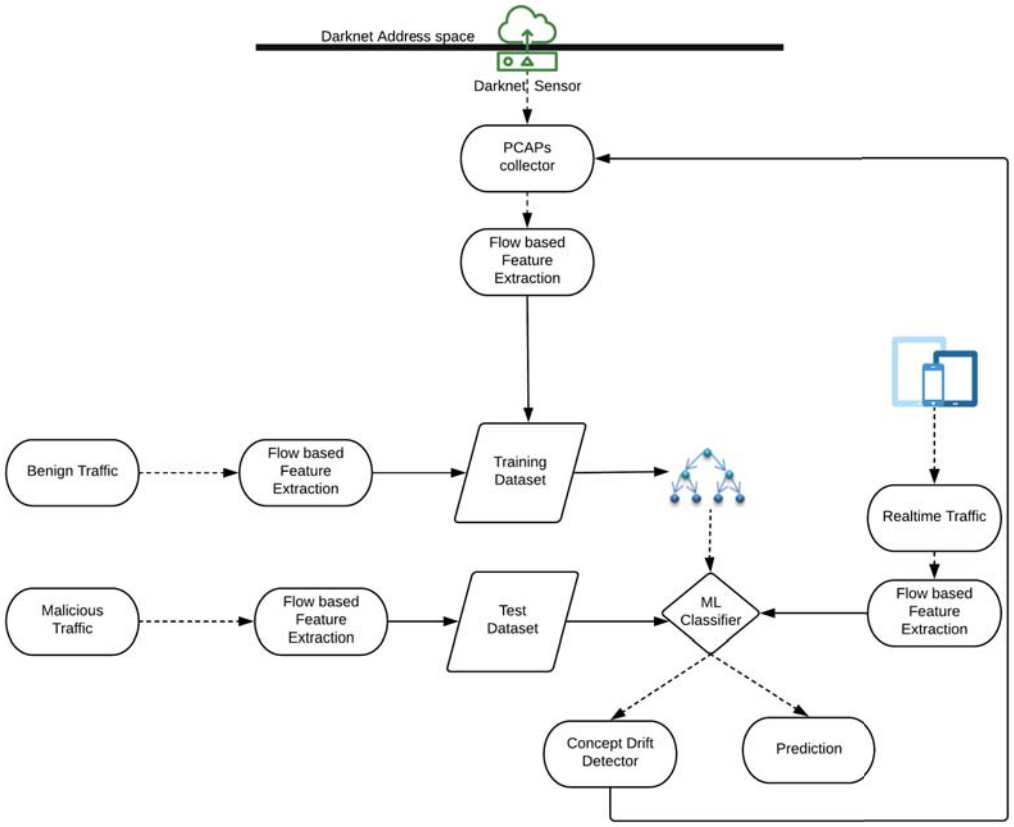


Figure (1) Darknet traffic based threat detection system

Each instance of the dataset was labeled either malicious or normal. The data was cleaned for missing values. We used combination of several feature scaling methods with different ML algorithms to perform experiments. Table IV shows the features scaling techniques and the ML algorithms used in this study.

D. Concept drift detector

The concept drift detector in our framework will trigger an alarm when the performance of the classifier is degraded by a certain threshold. It measures the magnitude of the drift by analyzing the new dataset. It also highlights the features which produce the drift. It verifies the need of new feature extraction or re-training of the classifier.

V. Experiments and Results

In this section, we discuss our experiment setup, evaluation metrics, hyper-parameter optimization and the experiment results.

A. Experiment setup

The darknet feeds were collected on a Virtual Machine (VM) provided by SURFnet. The feeds resided on this VM in the premises of SURFnet and the features from the feeds were extracted on that VM. The VM included a 8 core processor and 16 GB RAM. We used the Microsoft Azure ML platform to preprocess the datasets and to train the classifiers.

B. Performance evaluation metrics

We used the following parameters to evaluate the effectiveness of the ML classifiers.

1) Confusion Matrix: This shows the performance of the classification model. Rows in the confusion matrix represent the instances of the actual class, and the columns represent the predicted class.

TABLE (II) Confusion Matrix

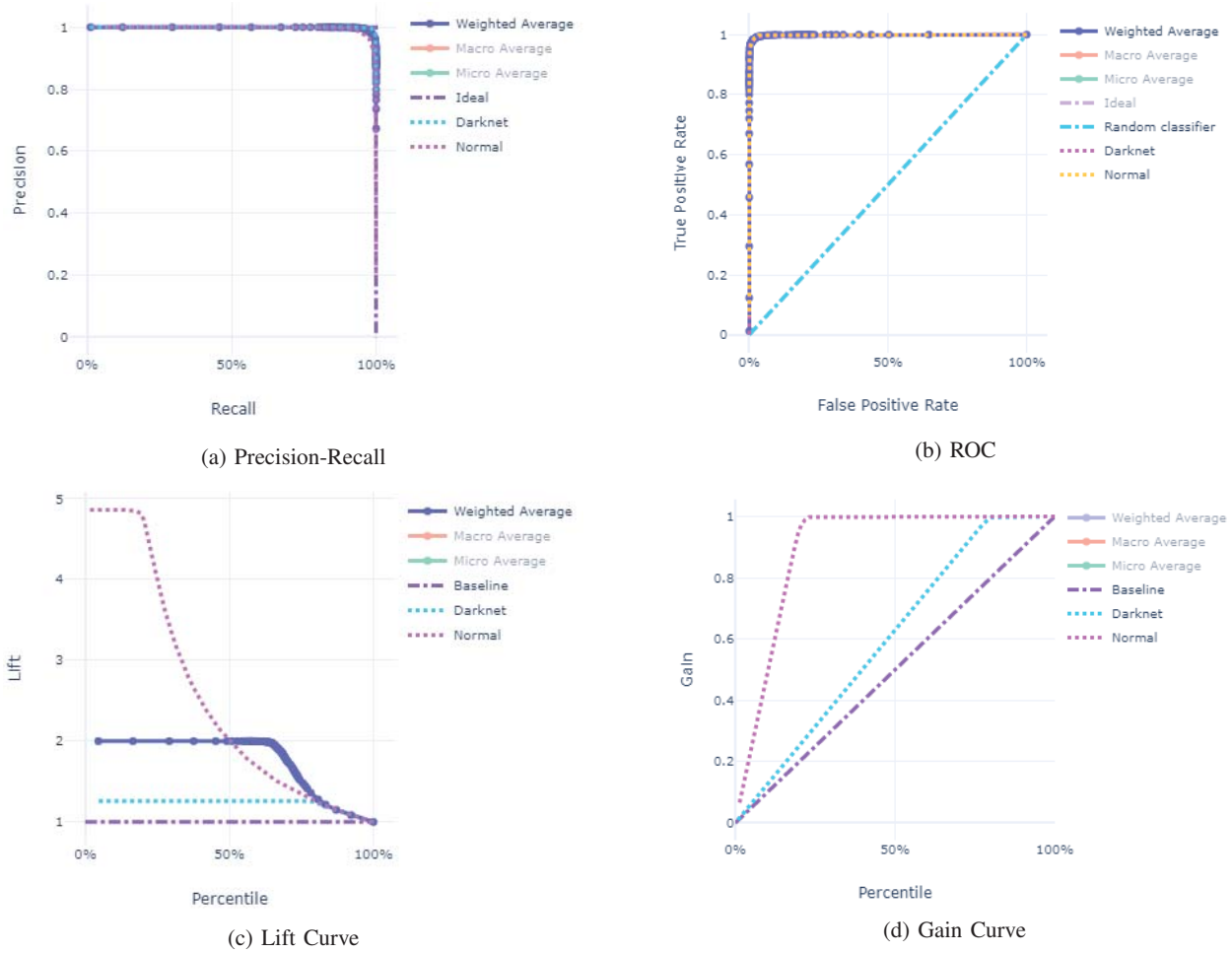
		Predicted	
		Darknet	Normal
Actual	Darknet	$TruePositive$	$FalseNegative$
	Normal	$FalsePositive$	$TrueNegative$

2) Accuracy: This is the percentage of the correctly predicted instances.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$$

3) F1 score: This is the harmonic mean of precision and recall.

$$F1Score = \frac{2*(TPR*Precision)}{TPR+Precision} = \frac{2TP}{2TP+FP+FN}$$



(a) Precision-Recall

(b) ROC

(c) Lift Curve

(d) Gain Curve

Figure (2) Graphs: Random Forest

4) Precision-Recall curve: This curve is useful to find the relationship between precision and recall. Precision represents the ability of a classifier to predict all instances correctly, while recall is the ability to find all the instances of a particular class.
5) Receiver operating characteristics (ROC): This is plot of the true positive rate and the false positive rate.
6) Area under ROC curve (AUC): This is the value derived from the ROC curve, and it tells how efficient the classifier will be on unseen instances.
7) Lift Curve: This curve shows the value gain of the particular model. It shows how much better one can expect with this model.
8) Gain Curve: This shows the performance of the model by the percentage of the dataset.

C. Hyper-parameter Optimization

To find the best set of hyper-parameters, we applied the following four steps:

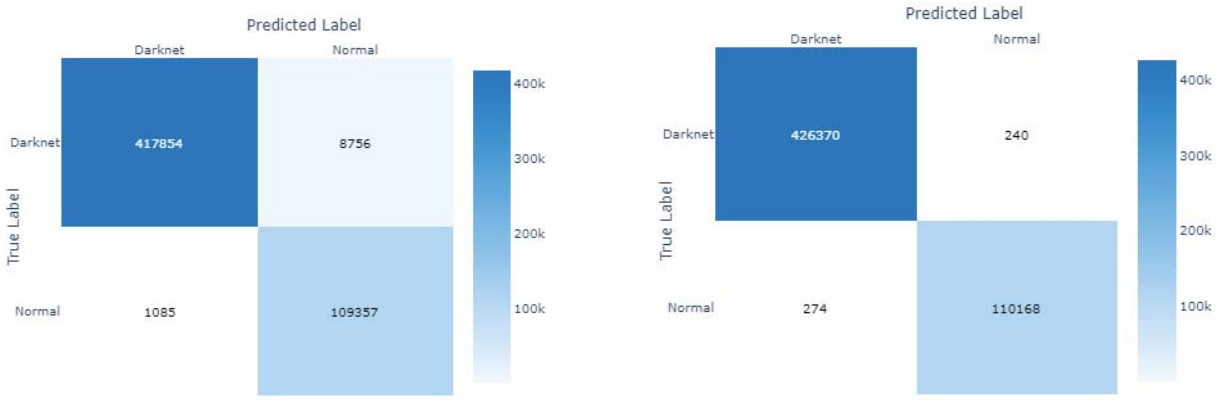
1) Define the exact parameter for the algorithm to consider.
2) Define the cross-validation setting and the number of folds for the dataset.
3) Decide on the metric for determining the best parameters. In our case, accuracy is chosen.
4) Train, evaluate, and compare the models. In this phase, cross-validation is performed on each set of parameter values.

D. Performance evaluation and Results

The experiments performed in this study were based on several ML algorithms, tuned at different hyper-parameter settings. The dataset was divided into training, validation and test set. The cross validation method was used in our scenario.

Table IV shows the individual accuracy and iterations of each algorithm. It can be seen in Figure 5 that the accuracy increases with the number of iterations.

From the ML algorithms that we evaluated during our experiments, we chose two best algorithms according to several criteria, such as training time, efficiency, memory usage, and capability of handling large-scale data. Table III shows the performance evaluation of these algorithms. Accuracy of around 98% was obtained by Random Forest (RF) and 99% by lightGBM. The table also shows the preprocessing techniques



(a) Confusion Matrix for Random Forest

(b) Confusion Matrix for LightGBM

Figure (3) Confusion Matrix

TABLE (III) Best models in terms of accuracy and hyperparameter optimization

Iterations	Preprocessor	Algorithm	Time	F1 Score	AUC	Accuracy
2	MinMaxScaler	RandomForest	0:01:54	0.98191	0.99908	0.98167
13	MaxAbsScaler	LightGBM	0:02:20	0.99904	0.99997	0.99904

TABLE (IV) Accuracy of the algorithms at various iteration levels

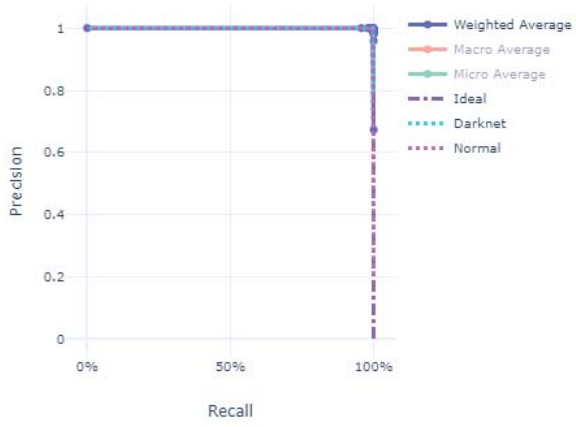
Iterations	Algorithm	Accuracy
13	MaxAbsScaler, LightGBM	0.99904
16	MinMaxScaler, lightGBM	0.99573
20	MaxAbsScaler, LightGBM	0.99470
19	SparseNormalizer, lightGBM	0.99378
15	StandardScalerWrapper, LightGBM	0.99334
22	StandardScalerWrapper, LightGBM	0.99235
18	RobustScaler, lightGBM	0.98984
21	StandardScalerWrapper, LightGBM	0.98830
14	TruncatedSVDWrapper, KNN	0.98630
2	MinMaxScaler, RandomForest	0.98167
0	StandardScalerWrapper, SGD	0.96905
3	StandardScalerWrapper, SGD	0.95788
11	StandardScalerWrapper, LogisticRegression	0.95734
1	MinMaxScaler, SGD	0.94716
8	TruncatedSVDWrapper, LogisticRegression	0.94294
12	TruncatedSVDWrapper, LogisticRegression	0.94117
4	StandardScalerWrapper, ExtremeRandomTrees	0.92591
6	MinMaxScaler, ExtremeRandomTrees	0.92219
5	MinMaxScaler, RandomForest	0.92036
7	SparseNormalizer, RandomForest	0.89899
23	StandardScalerWrapper, GradientBoosting	0.89114
10	StandardScalerWrapper, LogisticRegression	0.88748
9	MaxAbsScaler, RandomForest	0.88618

and time taken to build these classifiers. Figure 3 shows the confusion matrix obtained from these algorithms

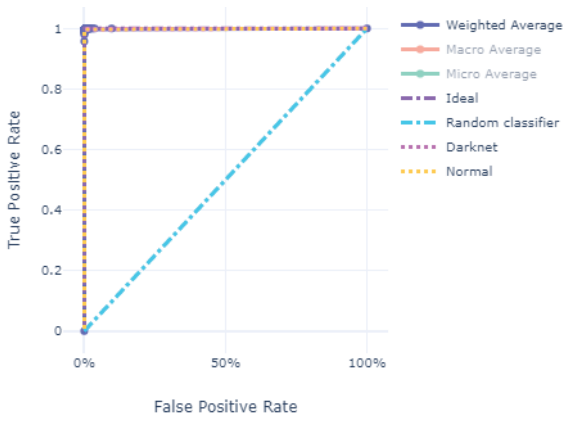
Figure 2 and 4 show the graphs of the performance evaluation obtained by these algorithms. The ROC curve and precision-recall plot show an ideal situation in both algorithms. The results of the lift and gain curve are also promising and show how much better the model can perform. The overall results from these graphs show that the model is properly trained and can predict the labels accurately.

VI. Conclusion

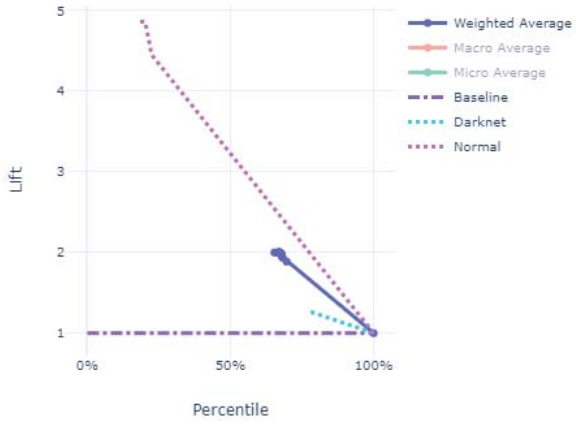
We presented a framework for threat detection using darknet traffic. By extracting the proper network traffic features of the threats in the darknet traffic, a pattern can be generated by ML algorithms, which can detect not only known threats but also unknown threats of the same kind. Proper feature extraction and hyper-parameter tuning play an important role in ML based threat detection. Also, concept drift is very common for ML based solutions and the ML model needs to be re-trained with the advent of time. The RF and lightGBT models with fine-tuned hyper-parameters produced the best results during the experiments. Future work is to extract more features from the traffic and combine the output of several ML algorithms to make optimized decisions. Furthermore, deep learning algorithms will be used in our future work.



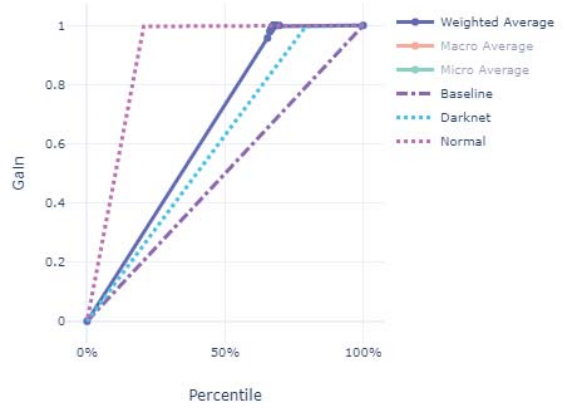
(a) Precision-Recall



(b) ROC



(c) Lift Curve



(d) Gain Curve

Figure (4) Graphs: Light GBM

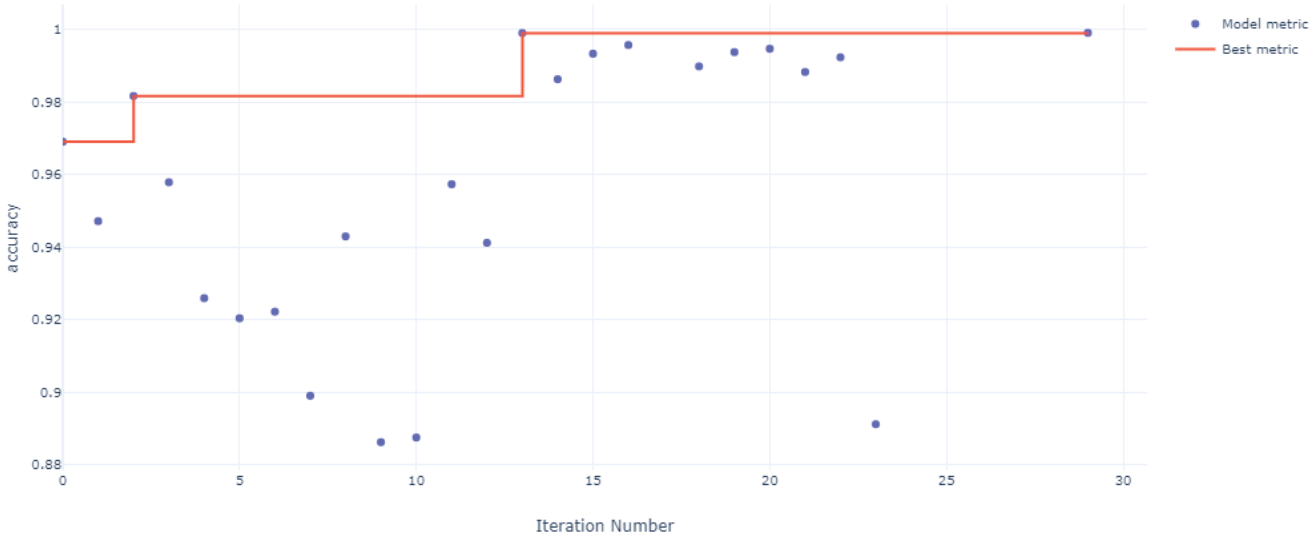


Figure (5) Iteration vs Accuracy Chart

References

[1] Juniper-Research. 'internet of things' connected devices to almost triple to over 38 billion units by 2020. [Online]. Available: https://www.juniperresearch.com/press/press-releases/iot-connected-devices-to-triple-to-38-bn-by-2020

[2] Kaspersky. How to not break the internet. [Online]. Available: kaspersky.com/blog/attack-on-dyn-explained/13325/

[3] Kaspersky-Lab. Hajime, the mysterious evolving botnet. [Online]. Available: https://securelist.com/hajime-the-mysterious-evolving-botnet/78160/

[4] Trend-Micro. Millions of networks compromised by new reaper botnet. [Online]. Available: https://www.trendmicro.com/vinfo/pl/security/news/cybercrime-and-digital-threats/millions-of-networks-compromised-by-new-reaper-botnet

[5] S. Kumar, A. Viinikainen, and T. Hamalainen, "A network-based framework for mobile threat detection," in *Proc. 1st Int. Conf. Data Intelligence and Security (ICDIS)*, Apr. 2018, pp. 227–233.

[6] S. Kumar, A. Faizan, A. Viinikainen, and T. Hamalainen, "Mlspd - machine learning based spam and phishing detection," *Computational Data and Social Networks*, Jan. 2018. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-04648-4_43

[7] T. Cymru. The darknet project. [Online]. Available: https://www.team-cymru.com/darknet.html

[8] ShadowServer. Darknet report. [Online]. Available: https://www.shadowserver.org/what-we-do/network-reporting/darknet-report/

[9] E. Bou-Harb, C. Assi, and M. Debbabi, "Csc-detector: A system to infer large-scale probing campaigns," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 3, pp. 364–377, May 2018.

[10] R. Zhang, C. Yang, S. Pang, and H. Sarrafzadeh, "Unitecdeamp: Flow feature profiling for malicious events identification in darknet space," in *Applications and Techniques in Information Security*. Springer, Jan. 2017. [Online]. Available: http://dx.doi.org/10.1007/978-981-10-5421-1_13

[11] I. Škrjanc, S. Ozawa, D. Dovžan, B. Tao, J. Nakazato, and J. Shimamura, "Evolving cauchy possibilistic clustering and its application to large-scale cyberattack monitoring," in *Proc. IEEE Symp. Series Computational Intelligence (SSCI)*, Nov. 2017, pp. 1–7.

[12] E. Balkanli, A. N. Zincir-Heywood, and M. I. Heywood, "Feature selection for robust backscatter DDoS detection," in *Proc. IEEE 40th Local Computer Networks Conf. Workshops (LCN Workshops)*, Oct. 2015, pp. 611–618.

[13] S. H. A. Ali, S. Ozawa, T. Ban, J. Nakazato, and J. Shimamura, "A neural network model for detecting DDoS attacks using darknet traffic features," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, Jul. 2016, pp. 2979–2985.

[14] N. Furutani, T. Ban, J. Nakazato, J. Shimamura, J. Kitazono, and S. Ozawa, "Detection of DDoS backscatter based on traffic features of darknet TCP packets," in *Proc. Ninth Asia Joint Conf. Information Security*, Sep. 2014, pp. 39–43.

[15] UNB. Cicflowmeter (formerly iscxflowmeter). [Online]. Available: http://www.netflowmeter.ca/netflowmeter.html

[16] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," 2017.

[17] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy, ICISSP 2016, Rome, Italy, February 19-21, 2016.*, O. Camp, S. Furnell, and P. Mori, Eds. SciTePress, 2016, pp. 407–414.

[18] Microsoft. Azure machine learning service. [Online]. Available: https://azure.microsoft.com/en-us/services/machine-learning-service/