2019 IEEE/IFIP International Conference on Dependable Systems and Networks Workshops

Design of a Classification Model for a Twitter-based Streaming Threat Monitor

Fernando Alves, Pedro M. Ferreira, and Alysson Bessani
LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
Email: {fbalves, pmferreira, anbessani}@fc.ul.pt

***Abstract*—Receiving timely and relevant security information is crucial to maintaining a high-security level on an IT infrastructure. This information can be extracted from Open Source Intelligence published daily by users, security companies, and hackers. In particular, Twitter has become an information hub for obtaining cutting edge information about many subjects, including cybersecurity. This work discusses the design of a classifier model for a Twitter-based threat monitor for generating a summary of the threat landscape related to a given monitored IT infrastructure. Since the classifier is a crucial element of the processing pipeline that constitutes the threat monitor, its architecture, topology and hyper-parameters must be properly selected to achieve high true positive and true negative classification rates. Our experimental work considered two architectural approaches: a single model for the whole IT infrastructure or an ensemble of models, one for each of several parts of the infrastructure. Within this scope we tested one linear (support vector machine) and one non-linear (multi-layer perceptron) modelling technique. Finally, several model design variables, hyper-parameters and learning parameters were selected by grid-search.**

***Index Terms*—Threat discovery; OSINT; Twitter; Machine Learning.**

I. Introduction

A security analyst must be aware of the latest developments regarding updates, patches, mitigation measures, vulnerabilities, attacks, and exploits to protect an infrastructure adequately. Awareness should raise within the *Security Operations Center* (SOC) through *Security Information and Event Management* (SIEM) software, thereby allowing correlating the latest information available with infrastructure events.

There are two primary ways of obtaining cybersecurity news. One is to purchase a curated feed from a specialised company such as SenseCy [2]. Another, is to collect *Open Source Intelligence* (OSINT) publicly available [25] from various sources on the internet (e.g., Threatpost [9]). Although there are numerous so-called *threat intelligence* tools (*e.g.*, SpiderFoot [8], IntelMQ [4], and AlienVault OTX [1]), their main focus is on collecting security-related OSINT from a wide variety of sources. At most, they apply simple keyword-based queries and filters to decrease the big volume of information but do not provide more elaborated processing or analysis. To overcome the limitations of keyword-based methods these tools have to be adapted or extended, configured, and possibly, complemented by the end user. However, recent work [17, 23, 29] demonstrates that different types of useful information and Indicators of Compromise (IoC) can be obtained from OSINT if more sophisticated analysis techniques are applied. These results highlight the gap between the current capabilities of existing OSINT-based tools and the potential that OSINT can raise.

To fill this gap, we considered the problem of designing an OSINT-based tool to keep SOC analysts aware of the threat landscape against the infrastructures under their responsibility. Besides choosing sources and collecting OSINT, this requires selecting only the information related to the security of the monitored infrastructure assets and joining identical information published by different sources. Given the volume of information, this is a time-consuming task for which security analysts have a limited time budget, even though the quality of their work depends on this awareness.

To this end, we developed a Twitter-based streaming threat monitor called SYNAPSE [10]. SYNAPSE's pipeline is composed of filtering, feature extraction, binary classification, aggregation, and generation of indicators of compromise. More specifically, an automated tool gathers tweets from security-related accounts, a supervised machine learning technique selects those relevant for the specified infrastructure being monitored, and a clustering method avoids presenting repeated or unnecessary information.

Twitter was chosen for two main reasons. First, Twitter is well-recognised as a relevant source of short notices (almost in real-time) about web activity and occurring events [3]. Second, the limited size of a tweet makes it simple to process through general-purpose machine learning approaches, which enable low error levels across multiple domains of application. Furthermore, although short, tweets provide enough elements to categorise their content, as well as links for more detailed material.

Previous work to gather cybersecurity information from Twitter also include machine learning techniques to deem information as relevant. Mittal *et al*. [19] use a knowledge base created from security concepts, to evaluate if a tweet is relevant for cybersecurity. Ritter *et al*. [20] search Twitter for occurrences of three specific *topics*: DoS attacks, data breaches, and account hijacking; tweets are selected through an expectation-regularization classifier [28]. Trabelsi *et al*. [26] *cluster* tweets by subject; threats not referred by NVD are considered novel, and handled like zero-day vulnerabilities. Le Sceller *et al*. [15] designed a framework that collects tweets on keyword basis, but is capable of extending the keyword set automatically. This framework employs various techniques, including clustering and local sensitive hashing [13]. Dionísio

978-1-7281-3030-9/19/$31.00 ©2019 IEEE
DOI 10.1109/DSN-W.2019.00010

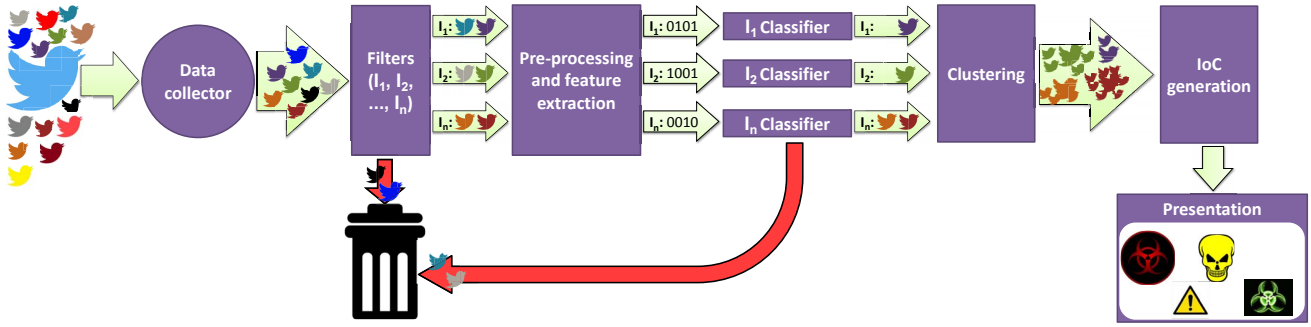

Fig. 1: The architecture of the Twitter-based threat monitor [10]. Collected tweets pass through the various stages and those classified as relevant are aggregated and delivered to SOC analysts.

et al. [12] used deep learning techniques to detect and extract security-related information from tweets. Finally, Sabottke *et al*. [23] show that information about exploits are published on Twitter (on average) two days before they are included in the NVD, selecting data using Support Vector Machines [11].

This paper focuses on selecting the architecture of SYNAPSE classification stage and on the design of the the classification models employed, to obtain high true positive and true negative classification rates. Besides model design variables and hyper-parameters, two architectural choices were considered: whether to employ a linear or a non-linear classification technique, and whether to describe the monitored infrastructure as a whole or divide its elements into smaller parts, each part having its own classifier. Interestingly, the linear model performed better when considering the whole infrastructure, while the non-linear one achieved better results as an ensemble. In both cases, the classifiers achieved True Positive and True Negative Rates (TPR and TNR) around 90%.

II. SYNAPSE's Pipeline

Figure 1 presents SYNAPSE architecture and data processing stages: tweet gathering, filtering, feature extraction, classification, clustering, and IoC generation. The following sections describe each stage.

A. Data Collection

The data collector module requires a set of accounts, from which it will collect every tweet posted using Twitter's stream API. These accounts can be from security analysts and companies, vendors, hackers, researchers, among others. They are chosen considering the likelihood of users tweeting about the security of elements belonging to the monitored IT infrastructure. Since most security analysts already follow OSINT sources (including Twitter accounts), it is just a matter of including them in the account set to tailor and improve OSINT coverage.

We opted to collect tweets from selected accounts instead of using a keyword-based approach since the latter is likely to retrieve large amounts of irrelevant information, an approach already in use in the literature [24]. For instance, tweets with the word "windows" include all Windows-related topics (the OS) and all tweets referring glass windows, besides other non-security related topics. By collecting tweets only from selected security-related accounts, a more substantial fraction of tweets are related to IT security, leaving the focus on filtering tweets not mentioning threats related to the specified IT infrastructure.

B. Filtering

Despite the account-based collection approach, most likely the collected data will include tweets unrelated to the infrastructure under the analyst's care. These have to be dropped by a filter. The filtering approach assumes that a tweet referring a threat to a particular IT infrastructure asset has to mention asset properties. Therefore, a second input is required: a set of keywords describing the assets of the monitored IT infrastructure. Only tweets that include at least one of the keywords will pass the filter. Keywords further restrict the scope of the security events, hence decreasing the number of irrelevant tweets beyond the filter.

To maximize the effectiveness of the tool, the keywords defining the monitored assets must be as complete as possible. Keywords can, in part, be learned from logs and network traffic within the infrastructure, but have to be complemented with slang and other informal terms commonly used by IT professionals. Although that work is out of the scope of this article, this process may be substantially automated. For example, if the analyst is in charge of securing a Linux cluster running virtual machines to serve a web service with a database, the keyword set could be {`linux, ssh, virtualbox, vbox, mysql, apache http, php`}. In principle, the more complete and specific the keyword set, the more effective the filtering process will be. For example, instead of using only "Linux", the analyst could also specify the specific distribution in use.

C. Pre-processing and Feature Extraction

Pre-processing aims to normalise the tweet representation. First, all characters are converted to lower case, and stopwords and hyperlinks are removed (these offer little information since they are shortened URLs). Numbers, dots, and hyphens are replaced by their textual representation (*e.g.*, "2" to "two"), as these are relevant to distinguish software versions (*e.g.*,

Mozilla Firefox 4.5.1-2). Finally, all non `[a-z]` characters are removed. For instance, after pre-processing, the tweet,

```
#Oracle #Linux 6 / 7 : Unbreakable
Enterprise kernel (ELSA-2016-3573)
https://t.co/vLTel8NodG #Nessus
```

becomes:

```
oracle linux six seven unbreakable
enterprise kernel elsa hyphen two
thousand and sixteen hyphen three
thousand five hundred and seventy
three nessus
```

Before classification tweets must be converted to a numerical format, thus becoming suitable for supervised learning binary classification techniques. This work uses the well-known Term Frequency - Inverse Document Frequency (TF-IDF) method [16]. TF-IDF computes weights to words (features) based on their occurrence frequency in each specific document and group of documents considered. The weight of a word increases with its frequency of occurrence in a single document but is scaled down by the frequency of occurrence in all documents. By mapping each consecutive word token to a corresponding vector position, tweets are converted to a constant size, zero-padded, TF-IDF numeric vector. Finally, to limit the size of the vector we employ the hashing trick technique [27].

D. Classification

For the classification of tweets according to their security relevance, two classifiers were tested: Support Vector Machines (SVM) [11] and Multi-Layer Perceptron (MLP) Neural Networks (NN) [21, 22]. The SVM is a broadly-used classifier achieving good results across a multitude of application domains. In this paper we consider the linear version of SVM. The MLP is a well-established and frequently used NN architecture that has a long track record of good and consistent results over a vast number of classification tasks.

E. Clustering

An aggregation phase should be included in threat intelligence tools to remove duplicate information and provide a concise summary of the current threat landscape. To perform such summarization we have proposed [10] a clustering algorithm. Clustering is a data-exploration technique designed to find groups of similar items within a set, and therefore is a natural choice for the problem of finding tweets discussing the same threats. SYNAPSE clustering stage relies on a stream clustering methodology supported by the *k*-means algorithm [18]. *k*-means is a a widely used algorithm that has provided good efficiency and empirical success over the last 50 years [14].

F. MISP Compatible IoC Generation

After the clustering phase, the clusters of tweets should be transformed into the IoC format to allow their inclusion in SIEMs or threat intelligence platforms. There are several standards for sharing IoCs, such as STIX [5] or MISP [6]. The format must be extendible and adaptable as tweets are unstructured and contain unpredictable content. For these reasons the MISP format has been selected to generate IoCs. Moreover, it can be easily converted into other standard formats like STIX.

To ease the correlation of the collected threats within the threat intelligence platforms, events should be categorized using tags, added according to a set of threat categories. The categories can be adapted from existing taxonomies, such as those from ENISA and VERIS for cyberthreats [7].

name	keywords
A	oracle, cisco
B	google chrome, chrome, internet explorer, firefox, microsoft edge, edge
C	wordpress, joomla, wp
D	microsoft windows, ms, linux, operating system, operating systems
ABCD	$A + B + C + D$

TABLE I: The infrastructure designed for tweet collection and filtering and its subdivision in four coherent parts.

III. Experimental Setup

In this section we describe the experimental work that was carried out design and validate SYNAPSE's classifier. The code is written in Scala using the Apache Spark Framework[1] pre-built with Hadoop. We chose Spark as their data-structures are scalable and designed for large datasets. Besides, Spark includes a scalable machine learning library called MLlib[2] including all ML-based algorithms used in this work.

A. Infrastructure definition

In a large organization the IT infrastructure is composed of many hardware and software assets. By using risk analysis the analyst selects a subset for which OSINT should be collected, filtered and summarized. The diversity of assets that may be selected in a large and complex organization, raises one question related to the classification stage: is it better to have one classifier covering the whole infrastructure being monitored, or is it preferable to have multiple classifiers focused on specific parts?

Considering this question, the hypothetical IT infrastructure designed for the experimental work was divided in four parts as presented in Table I. The table presents the keywords that are used in the filtering stage. The last row considers the case where one single classifier will be fed by tweets related to any of the four infrastructure parts.

Part `A` is a simple representation of Cisco and Oracle products, part `B` considers the browsers used in the organization, part `C` relates to content management systems deployed, and finally part `D` considers the operating systems used.

[1] http://spark.apache.org/

[2] https://spark.apache.org/docs/latest/mllib-guide.html

TABLE II: Datasets collection and labeling details.

Dataset:	D1		D2		D3	
Time period (from/to)	01/11/2015 01/04/2016		01/04/2016 15/05/2016		15/05/2016 10/07/2016	
Account sets	S1		S1, S2			
Num. tweets	71024		57579		66608	
	Pos.	Neg.	Pos.	Neg.	Pos.	Neg.
Related to A	556	514	177	249	502	256
Related to B	217	497	86	446	420	362
Related to C	486	606	138	900	425	303
Related to D	441	691	138	2697	336	1232
Rel. to ABCD	1697	2008	536	4292	1680	2153

TABLE III: Sets of accounts used to create the datasets.

S1 Accounts: inj3ct0r, TrustedSec, Anomali, briankrebs, Secunia, exploitdb, alienvault, slashdot, dstrom, Info_Sec_Buzz, vuln_lab, threatintel, dangoodin001, ivspiridonov, ThreatFeed, pikisec, SANSInstitute, johullrich, drericcole, F1r3h4nd, MaldicoreAlerts, USCERT_gov, gcluley, hal_pomeran, SecurityWeek, SecurityNewsbot, sans_isc, e_kaspersky
S2 Accounts: TenableSecurity, securitywatch, securityaffairs, zer0element, notsosecure, CyberExaminer, SCMagazine, DMBisson, lennyzeltser, IT_securitynews, teamcymru, WordPress, MicrosoftEdge, JoomlaTips, sjzaib, SecurityMagnate, Cisco, Dell, linuxtoday, securityninja, cyberopsy, OWASP_Java, _WPScan_, d_plusk, threatpost, Rootsector, Microsoft, linuxfoundation, ChidoDike, Sec_Cyber, ptracesecurity, msftsecurity, LinuxSec, hack3rsca, CiscoSecurity, NytroRST, joomla, Windows, crackerhacker00, fstenv, HPE_Security, googlechrome, wordpressdotcom, packet_storm, RokaSecurity, Oracle, firefox, wpbeginner, YoKoAcc, SecurityCrap, jasonlam_sec, threatmeter

B. Tweet collection

Three datasets were collected during three periods of time as shown in Table II, where the collection period, the sets of accounts used, the number of positive and negative tweets, and the distribution over the infrastructure parts, may all be observed.

After being collected the tweets were visually inspected and manually labeled as positive (mentioning a threat to a given part of the IT infrastructure) or as negative, thus creating labeled data sets suitable for supervised learning approaches. Four rows in Table II identify the numbers of positive tweets related to each of the infrastructure parts considered. The number of ABCD's positive tweets is less than the sum of the parts because tweets mentioning more than one infrastructure part were not duplicated when the tweets were merged.

As shown in the third row, two sets of accounts, S1 and S2, were used for tweet collection. The accounts per set are identified in Table III.

C. Feature extraction

We used Spark's implementation of TF-IDF using default parameters, except for the feature vector size. In order to find a suitable vector size to describe the tweets, eleven values were tested: $\{30, 50, 80, 100, 200, 300, 500, 750, 1000, 1500, 3000\}$.

D. Classification

As explained in Section II-D, two classifiers were employed: a linear SVM and an MLP Neural Network. In their design, relevant design variables and parameters were varied in order to find the best approach for this application.

	Configurations	A	B	C	D	ABCD
SVM	Feature size	3000	3000	3000	3000	3000
	Step size	0.5	0.05	0.2	0.01	0.05
	C	0.5	1.5	5.0	1.5	5.0
MLP	Feature size	1500	3000	3000	3000	3000
	Num. layers	4	5	3	7	5
	Neurons/layer	5	5	10	10	7

TABLE IV: The best configurations obtained for each classifier and dataset.

For the SVM we varied C, the regularization parameter, within $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5\}$, and the step size, a parameter for the Stochastic Gradient Descent (SGD) method used to train the SVM, within $\{0.1, 0.5, 1, 1.5, 2, 5\}$. For the MLP, we varied the number of layers from 2 to 8, and the number of neurons per layer within $\{5, 7, 10, 12, 14, 16, 18, 20\}$. The models were trained using dataset D1 and evaluated by performing 10-fold cross-validation. In both cases the limit on the maximum number of training iterations was set to 100.

IV. Results

Figure 2 shows Pareto curves for all the tested configurations. Each point shows average values obtained by the various configurations over the 10-fold cross-validation procedure. The Pareto front is shown with lines connecting the dominant configurations in terms of True Positive Rate (TPR—x-axis) and True Negative Rate (TNR—y-axis), for both types of classifiers. For infrastructure part A (oracle, cisco), it is possible to see that the SVM solutions dominate the MLP ones. A possible explanation is that this infrastructure generated a simpler dataset, whose patterns were easily captured by a linear classifier, but not complex enough to properly train the MLP NN configurations that may have over-fitted the data. For the other infrastructures, each of the techniques dominate one of the evaluation metrics (infrastructure parts B and C), or the SVM clearly dominates TNR while the TPR results are comparable (infrastructure part D and complete infrastructure).

The points highlighted by a larger circle in the top-right of the figure are the Pareto-optimal configurations presenting the best balance between TPR and TNR (smallest distance to the optimum). Table IV presents these configurations, revealing that there is a clear advantage in using high-dimensional feature vectors. Regarding other design variables or parameters, the results exhibit more variability.

The results presented next were obtained by the Pareto optimal solutions described in Table IV. For that purpose, SYNAPSE's classifier was evaluated by using datasets D2 and D3. These datasets have tweets posted after those in the training data set (D1) and include information posted by an additional set of accounts (S2), not considered in the training stage. This evaluation methodology embodies the ideas that in a real situation, after being trained, models will classify data from future events, and that over time new accounts may be added (or removed) to the system.

Considering that cross-validation was employed during the model selection work, it should be noted that the selected

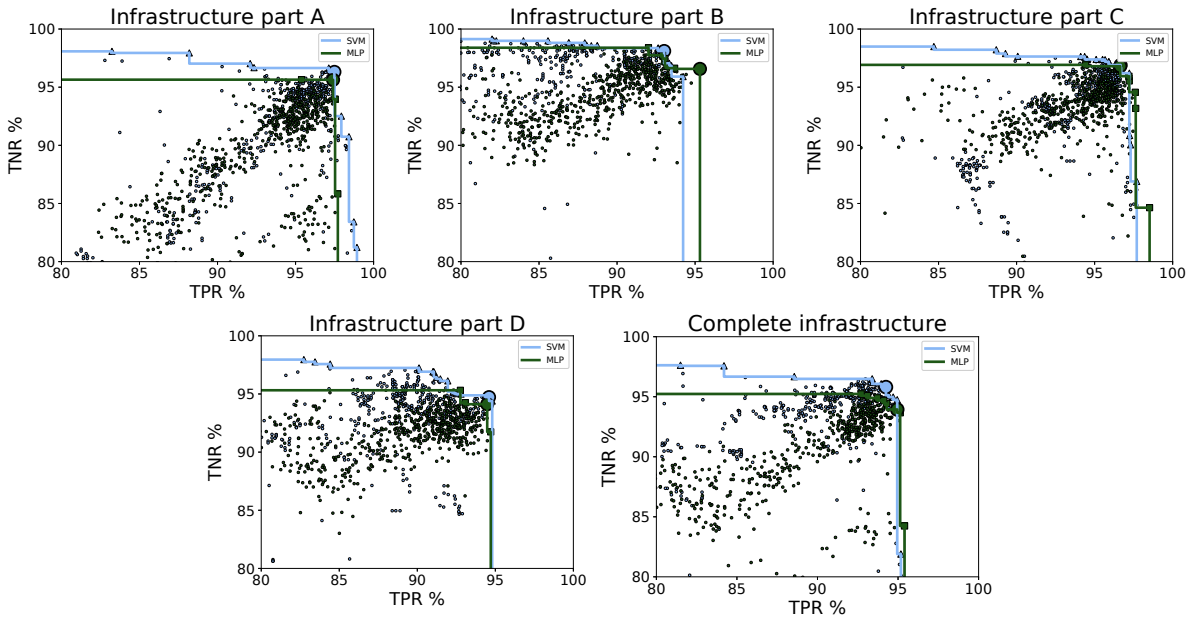

Fig. 2: The Pareto curves for SVM and MLP after 10-fold cross validation using dataset *D1*, for infrastructures A, B, C, D and ABCDE, respectively.

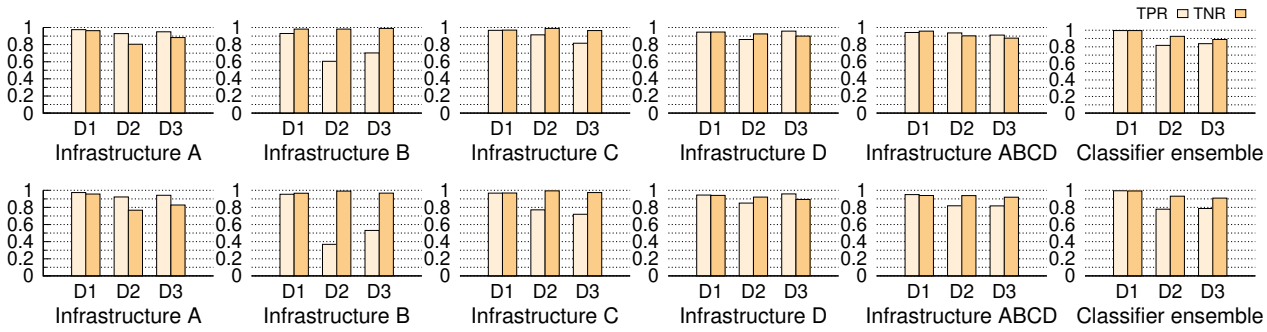

Fig. 3: SVM (top) and MLP (bottom) classifier results for infrastructures A, B, C, D, ABCD, and the classifier ensemble.

model configurations were retrained using the complete dataset D1. The feature vectors corresponding to tweets in D2 and D3 were generated using the TF-IDF model determined using dataset D1. This guarantees that TF-IDF weights attributed to words in D2 and D3 will be coherent with those employed to train the classifiers.

Figure 3 shows the performance of the best SVM and MLP classifiers measured in terms of TPR and TNR, considering also the average result obtained by 10-fold cross-validation with D1.

As expected, in general the results are slightly worse in D2 and D3 when compared to D1. This is the effect of new data that presents unmodeled patterns to the classifiers. As time passes and as new accounts are added this effect should be expected with increased impact.

Focusing on the results obtained in datasets D2 and D3, in general the classifiers maintain very high TPR and TNR except for the models specific to infrastructure part B, that exhibit a significant drop in TPR. This might be explained by the fact that part B has the smallest number of positive training examples (see Table II), hence it is more sensitive to the novelty of data in D2 and D3.

In most cases, the TNR is higher than the TPR, with exceptions for the models of infrastructure part A, where TPR is higher than the TNR, and for the MLP model of the complete infrastructure, where the results are comparable. In general a higher TNR might be explained by the imbalance between positively and negatively labeled data in the training data sets, which favors the TNR. The exception for infrastructure part A may be due to the simplicity of tweets collected for this part.

When comparing SVMs and MLP NNs, considering the specific models for the infrastructure parts, the results are comparable in terms of TNR, but show a consistent advantage of SVM models regarding the TPR. As a consequence this is also true for the SVM model ensemble, which is, for the SVM approach, the best solution. Regarding the classification

models trained for the complete IT infrastructure, the MLP NN achieved the best results, showing also the best balance between TPR and TNR.

Overall, the results of the SVM ensemble model standout, achieving high TPR and TNR classification rates, the best balance between these metrics, and little degradation from training to validation results.

V. Conclusions

This paper presents the experimental work carried out to select the classification approach used in a streaming Twitter-based threat monitor developed for threat awareness in security operation centres. The threat monitor, named SYNAPSE, gathers tweets from a set of Twitter accounts, filters them to target solely the monitored infrastructure, classifies remaining tweets as either relevant or not, aggregates tweets related to same threat using a stream clustering approach, and generates indicators of compromise suitable for threat sharing platforms. The results obtained from the model design experiments revealed that using a single classification model for the complete monitored infrastructure is preferable to using an ensemble of models for different infrastructure parts. Furthermore, when compared to multi-layer perceptron neural networks, support vector machines achieved the best balance between true positive and true negative classification rates. For the selected support vector machine model, these rates were very close or above 90%, showing little degradation from the training to the validation data sets.

References

[1] AlienVault OTX, The World's First Truly Open Threat Intelligence Community. https://otx.alienvault.com/. [Accessed 13-02-2019].

[2] Cyber Intelligence — SenseCy. https://www.sensecy.com/. [Accessed 13-06-2018].

[3] How people use Twitter in general - American Press Institute. https://www.americanpressinstitute.org/publications/reports/survey-research/how-people-use-twitter-in-general. [Accessed 13-06-2018].

[4] IntelMQ. http://github.com/certtools/intelmq/. [Accessed 13-06-2018].

[5] Introduction to STIX. https://oasis-open.github.io/cti-documentation/stix/intro. [Accessed 13-06-2018].

[6] MISP data models. http://www.misp-project.org/datamodels/. [Accessed 13-06-2018].

[7] MISP taxonomies. http://www.misp-project.org/datamodels/. [Accessed 13-06-2018].

[8] SpiderFoot, Open Source Intelligence Automation. http://spiderfoot.net/. [Accessed 13-06-2018].

[9] Threatpost — The first stop for security news. https://threatpost.com/feed/. [Accessed 13-06-2018].

[10] Fernando Alves et al. Processing tweets for cybersecurity threat awareness. *CoRR*, abs/1904.02072, 2019.

[11] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3), 1995.

[12] Nuno Dionísio et al. Cyberthreat Detection from Twitter using Deep Neural Networks. In *Proceedings of the International Joint Conference on Neural Networks*, 2019. To appear.

[13] Aristides Gionis et al. Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529, 1999.

[14] Anil K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.*, 31(8), 2010.

[15] Quentin Le Sceller et al. Sonar: Automatic detection of cyber security events over the twitter stream. In *Proceedings of the 12th International Conference on Availability, Reliability and Security*, 2017.

[16] Jure Leskovec et al. *Mining of massive datasets*. Cambridge University Press, 2014.

[17] Xiaojing Liao et al. Acing the IOC game: Toward automatic discovery and analysis of open-source cyber threat intelligence. In *Proc. of the 23rd ACM CCS*, 2016.

[18] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th BSMSP*, 1967.

[19] Sudip Mittal et al. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In *Proc. of the 8th IEEE/ACM ASONAM*, 2016.

[20] Alan Ritter et al. Weakly supervised extraction of computer security events from twitter. In *Proc. of the 24th WWW*, 2015.

[21] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 68(6), 1958.

[22] David E Rumelhart et al. Learning internal representations by error propagation. Technical report, DTIC, 1985.

[23] Carl Sabottke et al. Vulnerability disclosure in the age of social media: exploiting twitter for predicting real-world exploits. In *Proc. of the 24th USENIX Security Symp.*, 2015.

[24] Anna Sapienza et al. Discover: Mining online chatter for emerging cyber threats. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, 2018.

[25] Robert David Steele. Open source intelligence: What is it? why is it important to the military. *American Intelligence Journal*, 17(1), 1996.

[26] Slim Trabelsi et al. Mining social networks for software vulnerabilities monitoring. In *Proc. of the 7th NTMS*, 2015.

[27] Kilian Weinberger et al. Feature hashing for large scale multitask learning. In *Proc. of the 26th ICML*, 2009.

[28] Mohammed J Zaki et al. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[29] Ziyun Zhu and Tudor Dumitras. FeatureSmith: Automatically Engineering Features for Malware Detection by Mining the Security Literature. In *Proc. of the 23rd ACM CCS*, 2016.