CTI-Twitter: Gathering Cyber Threat Intelligence from Twitter using Integrated Supervised and Unsupervised Learning

Linn-Mari Kristiansen*, Vinti Agarwal †, Katrin Franke *, Raj Sanjay Shah †

* Norwegian University of Science and Technology, Gjovik, Norway

†Birla Institute of Science and Technology, Pilani, India

2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9378393

***Abstract*—Cyber threat intelligence (CTI) can be gathered from multiple sources, and Twitter is one such open source platform where a large volume and variety of threat data is shared every day. The automated and timely mining of relevant threat knowledge from this data can be crucial for enrichment of existing threat intelligence platforms to proactively defend against cyber attacks. We propose CTI-Twitter: a novel framework combining supervised and unsupervised learning models to collect, process, analyze and generate threat specific knowledge from tweets coming from multiple users. CTI-Twitter has multi-fold contributions: i) first collecting tweets through Twitter API, ii) extracting relevant threat tweets from irrelevant ones, and classifying relevant ones into multiple classes of threats iii) then grouping tweets belonging to each class using topic modeling iv) finally performing data enrichment and verification process. We evaluate our proposed model on real-time tweets collected for about four months (in year 2020) using Twitter API. The encouraging results obtained indicate the effectiveness of CTI-Twitter in terms of timeliness and discovery of trending attacks patterns, and vulnerabilities.**

I. Introduction

With the growing rate of cybercrime, several organizations are struggling to get timely and relevant intelligence on cybersecurity threats and vulnerabilities. However, as the knowledge is contained in a variety of sources, it needs to be extracted and presented to security analysts in a form that is easy to understand and well explainable. In some of these sources threat knowledge is available in a structured format such as National Vulnerability Database, CERT alerts etc. Whereas in others such as blog posts, Twitter, Reddit, dark web etc, information is highly unstructured and noisy, but are frequently updated.

This article focuses on a methodology to extract not only timely CTI from Twitter, but also the most informative tweets that can support operational, tactical as well as strategic decisions within a organization.

Some earlier work [3] on information extraction explored the dark web, which can be hard to collect data from. Twitter is open, free to use and has one of the largest userbases in the world. The tweets that share and discuss new strains of malware, zero-day attacks, analysis of current attacks, new vulnerabilities and so on can be gathered to extract intelligence. Generally, the number of posts of interest to security experts are heavily outnumbered by irrelevant posts. Therefore, creating an end-to-end model that gather and filter out the irrelevant tweets and organize knowledge from tweets can be beneficial to generate threat intelligence from a digital investigator or security analyst point of view.

However, a sufficient amount of research already existed that explored the possibility of using Twitter as a potential asset for cyberthreat awareness. Most of it focused on designing tools capable of performing binary classification of tweets either using keyword search [10], support vector machine (SVM) [3] or a deep learning classifier[5]. Later, Semantic Web and Named Entity Recognition (NER) models are applied to extract intelligence from the relevant tweets.

However, as these methods have proved much success in the enrichment of threat intelligence platform, they suffer in finding or investigating a particular category of cyber attack. For example, they filter relevant posts from irrelevant, but does not group tweets together to get an overview of a specific threat. That makes it difficult for security experts to extract all the necessary information related to a specific type of attack and attacks similar in terms of intention, behavior and targets.

In this article we formulate threat intelligence extraction from Twitter as a new research problem, which aims to facilitate security practitioners/investigators to find threat oriented knowledge from a large and rapidly increasing pool of content from Twitter. To achieve this, we present CTI-Twitter, which first collects data from Twitter using the official API and preprocesses the raw tweets into a well-structured format to facilitate learning tasks. Next, we apply a fine-tuned pre-trained classifier which labels tweets as relevant if they contain security-related information, otherwise irrelevant. Then security relevant tweets

are classified into one of the five classes (i.e. *INFO, ATK, MAL, CVE and MISC*). Details of these classes can be found in Section 3.

In the end, semantic clustering is performed on each class, to group semantically similar tweets. It helps analyst to discover the most relevant information about an attack and other similar attacks. Semantic clustering also helps to discover the trending keywords for an attack category and use them for data enrichment as well as collecting more tweets via targeted keyword search.

To perform the classification, we use BERT [4], a Bidirectional Encoder Representations from Transformers that has achieved state-of-the-art performance on various natural language processing (NLP) tasks. A significant advantage of using BERT in comparison to other contextual models (such as word2vec, GloVe) is that it is bidirectional i.e. learns the context of each word in a sentence using both previous and next context [3]. This allows the model to gain more context to the predicted words than the unidirectional models. Another reason for choosing BERT model in our work lies in its transfer learning ability to fine-tune on our data just by adding one additional output layer. For semantic clustering, two grouping techniques, K-Means and Latent Dirichlet Allocation (LDA), are explored to see what insights one can gain from each of them. Specifically, we address the following research questions in this article:

RQ1: How well does the BERT model perform in classifying tweets into binary as well as multi-classes?

RQ2: How meaningful are the clusters formed by semantic clustering in order to gain threat specific knowledge?

RQ3: How effective is CTI-Twitter for data enrichment in the existing threat intelligence platform?

The remainder of this paper is organized as follows. In Section II, we describe related literature in the field of study. Section III details the various phases of CTI-Twitter. Experiments performed to evaluate the proposed method are elaborated in Section IV. In the final section, the conclusion and future work are discussed.

II. Related Literature

CTI collection and processing from open sources on the internet is not a new topic. This section will discuss existing literature surrounding this topic.

A. Collecting Intelligence from Twitter

Park et al. [12] compared structures of the Twitter and Youtube social networks, and found that Twitter networks can be seen as loosely connected hub-and-spoke networks. By this, we can infer that there are often key users (hubs) providing information to many other users (spokes).

In [1], Al-Khateeb et al. discussed deviant groups on social media, e.g. deviant collective hackers. They found out that some anonymous hacker groups create specific hashtags such as "TangoDown" to spread activity on Twitter during an operation as a rallying call. He discovered that identification of these hashtags could help in tracking down information about the attack.

Mackey et al. [9] used biterm topic model first to isolate those tweets clusters associated with illegal online marketing and sales. Thereafter analyzing hyperlinks associated with these tweets to assess the characteristics of illegal online sellers.

Nuno et al. [5] developed a deep learning tool to collect CTI from Twitter relevant to assets in an IT infrastructure. Unlike open keyword search, they focused on collecting data from specific accounts related to monitored IT infrastructure which reduces the number of irrelevant tweets. They found that vulnerabilities might be discussed on Twitter before they are put into the national vulnerability database. Similarly, Sapienza et al.[17] used Twitter among other sources to generate cyber threat alerts, and found that the ransomware NotPetya was reported on Twitter months before being covered in mainstream media.

B. Natural Language Processing on Tweets

1) BERT: Recently, a BERT classifier has been explored in disaster classification by Ma et al. [8]. By using error analysis, they observed that the model may struggle with tweets that are too short, use sarcasm and metaphors, misleading hashtags and non-ASCII words. Zhu et al. [18] used BERT on Twitter data to detect offensive content in tweets. They compare the BERT-base version performance with a n-gram SVM classifier. The results indicated that the uncased version of BERT worked very well for classifying tweets and achieves better F1-score than the n-gram SVM. However, the n-gram SVM achieves a slightly better accuracy.

2) Other Models: In CyberTwitter [10], NER models are used on Twitter data to create an automatic vulnerability alert system, where a tweet that contains at-least two security terms were considered as relevant. Le et al. [6] implemented a classifier for CTI-relevant tweets using Common Vulnerabilities and Exposures (CVE) descriptions. They discovered that in lots of vulnerability posts the CVE keyword is missing, however description matches a known CVE pattern. This indicates that broader keywords are most useful to collect information, since searching with the keyword "CVE" might skip many important posts.

III. Proposed System

In this work, we develop a system, CTI-Twitter, to extract threat specific intelligence from raw tweets. This section discuss the overall architecture of the proposed system CTI-Twitter (as shown in Figure 1) and details about the stages involved in it. It is a 4-stage methodology, which consists of:

- **Data collection:** This stage involves exploration of different ways to collect data from Twitter and find the most effective approach to extract meaningful information without paying premium API charges.

- **BERT Classification model:** This is the central stage of our model, which first extracts the relevant tweets from the dataset and subsequently divide them into multi-classes using the BERT sequence classifier. In this article, three different BERT embedding approaches (base, large and Distil) are explored.
- **Semantic Grouping:** To discover the trending keywords and categorize semantically similar security tweets in the MAL and ATK classes, LDA and k-means clustering is performed.
- **Feedback mechanism** for data enrichment, such as finding more specific information based on keywords gathered from the grouping stage.

More details about data collection can be found in Section IV. The discussion of the remaining stages, BERT, semantic grouping and data enrichment, are given below.

A. BERT Sequence Classifier

Pre-trained language models have shown a great success in solving a variety of NLP tasks such as text classification, entity recognition, question-answering etc. In two ways, pre-trained text representations can be used for solving these tasks: 1) feature-based, where task-specific architectures are designed and pre-trained representations are used as additional features. 2) fine-tuning, which adds additional layers at the output and fine-tunes all pretrained parameters on the data. It reduces the need of heavily engineered tasks specific architectures.

BERT is the first fine-tuning representation model that has been pre-trained on the large Brown Corpus and the English Wikipedia [5]. This allows BERT to perform well even when only a small amount of training data is available for fine-tuning. It consists of an attention mechanism that learns contextual relationships in texts. Instead of considering a particular direction of a sentence, e.g. left-to-right or right-to-left sequence in a sentence, BERT takes in the entirety of the sentence and is therefore known as a Bidirectional Model.

BERT captures contextual information using two techniques: Mask Language Model and Next Sentence Prediction. In the Mask Language Model (MLM), BERT masks some of the words of the input and then tries to predict the masked words based on the context obtained from the non-masked words in the sequence. Whereas, in the Next Sentence Prediction, the model takes in a pair of sentences as the input and predicts whether the second sentence in the pair is the next sentence of the first sentence.

Base, Large and Distil: There are many variants of BERT which are built similarly, but has some key differences in number of layers, self-attention heads and dimensions. In this work we focus on three of them. The first is BERT base, which consists of 12 attention heads and 110 million parameters. The second is BERT Large that has 24 attention heads and 336 million parameters.

Victor et al. [16] proposed a smaller general-purpose pre-trained version of BERT known as DistilBERT. It consists of 12 attention heads and 66 million parameters which is a 40% reduction in size from the BERT base model. Despite its small size, it retains 95% of the performance of BERT. It is built on the idea that output distributions of large networks can be approximated using smaller networks.

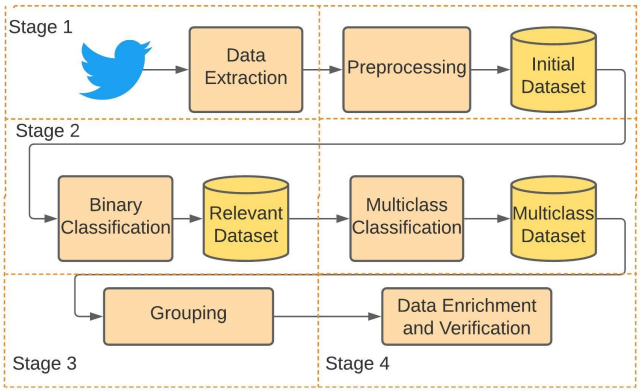

Fig. 1: System Overview

BERT vs Traditional pre-trained embeddings In traditional pre-trained text embeddings models such as Word2vec and Glove, vector representation for each token are learned using various contexts e.g. a single word can have multiple meanings in the text. However, as different contexts carry different meanings, and the final embedding is computed as average of all the vectors learned from different contexts for that particular token, the meaning of individual contexts are lost. On the other hand, BERT has context dependent and instance specific embeddings i.e. it does not provide representation of words, rather the representations of sub-sentences and sentences, which ensures that the significance of different contexts of the same word are retained.

1) Filtering Security relevant Tweets: To address RQ1, we manually classified some tweets randomly selected from the collected dataset to train a binary classifier. For a tweet to be considered relevant, the criteria used in our work is based on security reports and blogs that describes a list of indicators value for cyber investigations. For instance, if a tweet describes details about a cyber attack, target organization, identity of a threat actor, a new malware or vulnerabilities, we consider it to be relevant. An example is *Tweet 1: 'MegaCortex Ransomware is n** a**e to change Windows Password #MegaCortex#CyberAttack #RansomWare #Virus #Network'.* Since the tweet [1] describes a new version of the malware, we labelled it as 'relevant'.

Another example is: *Tweet 2: 'Protect Y**rse*f F**m a Cyber Attack'.* This does not give any information related to a cyber attack, so the warning is considered as 'irrelevant'.

[1]In the entire article, wherever tweets are shown as example, some text is kept hidden due to Twitter's privacy policy.

2) Tweets classification into multiple classes: We aim for classifying user tweets into five categories, which are given below.

INFO Informative posts, news and general vulnerabilities (not CVEs).

ATK Posts discussing cyber attacks.

CVE Posts discussing CVEs.

MAL Posts with details about malware.

MISC Posts that do not fit into the other categories. Can be commercials or news that are not directly related to cybercrime, or opinions and speculation.

From the previous example, *Tweet 1* will be classified as the MAL category. Another example for CVE class is: *'CVE-2011-36** An issue exists in Vanilla Forums before 2.*.1*.9 due to the w** cookies are handled'*.

The rationale behind multi categorization is so that security experts can determine different priorities to prevent damage to systems from malware attacks or stop the attacker from exploiting a vulnerability. Other tools such as [5] train the model to classify tweets into relevant and irrelevant. In our case, we make a more specific classification of relevant tweets, whereas the irrelevant ones are discarded. Most of the wrongly classified samples from the binary classification will be put into the MISC category by the multi-class classifier. Therefore, MISC will be the least helpful with a high variation or noise.

B. Performance Metrics

To measure the performance of the discussed classification models, we used several measures i.e. precision, recall and F1-score. Details of them are given in Appendix A.

C. Semantic Grouping

In order to identify groups of related tweets (e.g., those reporting new malware attacks), we cluster tweets belonging to the same category (e.g., those in *MAL class*). Clustering is needed for two reasons: (i) security analyst having hundreds of tweets in a specific category could experience information overload, wasting almost all advantages provided by the tweets classification, and (ii) knowing the details of IoCs related to a specific malware (e.g. *Covidlock*) can be important information for organization to prevent high damage.

We only cluster tweets classified as *ATK, MAL and INFO*, since only for such categories we could have a benefit identifying different groups. Indeed, tweets belonging to the *MISC* category are not informative, so they do not need to be analyzed at all.

Two text clustering methods are used to group semantically similar tweets: i) k-means with TF-IDF (Term Frequency-Inverse Document Frequency) and Sentence Transformer embeddings [14], and LDA [2].

k-means: is one of the most commonly used techniques for clustering. It starts by initializing the k cluster centers and input vectors (in our case tweets embeddings) are assigned to one of the existing clusters by computing the euclidean distance from the clusters. In every iteration, the centroids of the clusters are updated by computing the mean of each cluster and the input vectors are reassigned. The process continues until the cluster centers do not change. The major challenge of k-means is to find the optimal number of clusters. The elbow method and silhouette score are used to find the optimal value of k, where the metric used is the within-cluster sum of squares.

Two embedding approaches are used to generate input vector for k−means: TF-IDF[13] and sentence embeddings from the Sentence Transformer models [14]. The Sentence Transformer network is a modification of the BERT model using siamese and triplet networks that is capable of generating meaningful sentence embeddings such that sentences with similar meanings are close in vector space. This makes it popular in information retrieval and clustering tasks.

Latent Dirichlet Allocation (LDA): is another unsupervised learning method used to groups semantically similar documents in large corpora. LDA assumes that each documents in the corpus are derived from a generative process such that each document is a distribution of a finite set of topics, and each topic is a multinomial distribution of the vocabulary of words and each word of the document is drawn from one topic in the generative process.

In this work, each tweet is assumed as a document, which is converted into TF-IDF vectors for topic modeling in LDA. TF-IDF was set with a max document frequency of 0.75, to avoid generating topics from frequently used words. The stop words are chosen from common English words using the sklearn feature extraction library. Duplicate tweets are also removed from the dataset. More details about LDA can be found in [2].

D. Feedback mechanism

The output from our methodology can help security investigators with data enrichment for CTI in multiple ways:

1) Using trending keywords from LDA for query construction to gather more relevant information about a particular threat
2) Discover threat activity over a period of time
3) Discover a new set of indicators of compromise and identification of TTPs

We will perform data enrichment on identified keywords from LDA in this stage, to see what can be learned from this approach.

IV. Experimental Evaluation

A. Twitter Data Collection and Preprocessing

In this work, the data collection from Twitter is done in two phases i) the real-time and ii) the historical collection. The major results in this work are produced from the real-time collection. The historical tweets collection is done after discovering trending keywords from semantic clustering phase of CTI-Twitter. The real-time data is collected

TABLE I: Before and after data preprocessing

Before	After
Non-patching s**** again. It's not just OS' that n**d attention. #infosec #CyberSecurity https://t.co/xxxxxxx	non patching s**** again it not just os that n**d attention infosec cybersecurity
RobbinH**d #ransomware a** why it's increasing in popularity. https://t.co/xxxxxxx	robbinh**d ransomware a** why it increasing in popularity

for about four months using Twitter streaming API and has in total 76047 tweets. The historical collection is done in the data enrichment process, where targeted keywords are used. This collection method uses the GetOldTweets python library and is used since it can go further back in time than the free API.

1) Keywords selection: The set of keywords used by the streaming API are displayed in Listing 1. Some of the terms, like "malware" and "cve", resulted in a large number of posts, in contrast to specific terms such as "sqli". The information extracted using specific terms results in more relevant tweets and requires less filtering. However, this is not a viable approach for new emerging threats that may use terms that are either not used before or unknown to the investigator.

Listing 1: Keywords

```
keywords = ['malware', 'rootkit', 'cve',
    'shellcode', 'sqli', 'keylogging', '0
    day',
'cyber attack', 'keylogger', 'password
    cracker', 'metasploit', 'reductor',
'zombie proxy', 'C&C server', 'SQL
    injectable', 'leaked credentials',
'remote root', 'ransomware', 'security
    exploit']
```

2) Preprocessing: CTI-Twitter preprocesses the collected raw tweets by (i) filtering out other attributes *(such as usernames, created_at, followers count)* extracted along with tweet text, and then (ii) preprocessing the Text attribute of the tweet instance. Most tweets are assumed to be written by humans or bots simulating natural language. This creates the need for natural language processing to help the machine learning models interpret the information better. The preprocessing of the tweets is performed using the following steps.

- Tokenize text, and remove all the special characters, URLs, punctuation, and single characters.
- Convert words into lowercase and eliminate extra white-space along with stop words.

Some example tweets before and after preprocessing are shown in Table I.

After preprocessing, the summary of the data is described in Table II.

TABLE II: Dataset statistics

#Tweets/#Words per tweet	Count
Number of tweets (total)	76,047
Number of tweets manually labelled (into 2 classes)	2400
Number of tweets manually labelled (into 5 classes)	1220
Average number of words per tweet	13.27
Maximum number of words per tweet	43

B. Performance Evaluation and Model Selection

1) Binary Classification : To address the RQ 1, we first train the BERT binary classifier that can automatically assign each tweet the unique class labels {relevant or irrelevant}, as described in section III. First, we manually labeled a set of 2,400 tweets randomly selected from 76,000 tweets as either relevant (R) or irrelevant (I).

In total, of the considered 2,400 tweets we labeled 962 as relevant and 1438 as irrelevant. The number of samples which are used for training the models is 2400, where 1/5 of the samples are used for testing in each fold. The classification of the testing samples is used to calculate the performance measures.

Three variants of BERT model, BERT-Base, BERT-Large and BERT-Distil, are used to learn embeddings from text. The BERT sequence classifier we use for classification requires sentences of uniform length. The max length of sentences found in our dataset is 43. However, the size of tweets to be collected in future can be longer than what we have now. Therefore, we set the maximum sentence length up to 60. The shorter tweet sentences are padded with 0s to make the length uniform.

We also compare the BERT results with a CNN classifier [2], proposed in [5], to classify tweets into relevant and irrelevant and evaluate the overall performance by using k-fold cross validation. The CNN classifier generates a d-dimensional numeric vector to represent the semantic meaning of each word in a sentence using three different approaches i) by randomly initializing the values ii) using pre-trained word2vec embeddings and iii) using pre-trained GloVe embeddings.

Performance comparison: The CNN and BERT classifiers are implemented using TensorFlow (with default set of parameters) and Transformers library [3] respectively. The parameters used for implementation are listed in Table III. The AdamW [4] optimizer is used to train the parameters in the BERT model. The classification results for 5-fold and 2-fold cross validation are presented in Table V(a) and Table IV(a). The optimal number of epochs where SequenceClassifier balances both training

[2]Due to Twitter's policy, tweets are not published by authors[5], we run the classifier on our dataset.

[3]https://huggingface.co/transformers/model_doc/bert.html

[4]https://huggingface.co/transformers/main_classes/optimizer_schedules.html

TABLE III: List of parameters used in BERT SequenceClassifier and CNN

Parameters	BERT SequenceClassifier		CNN	
	Binary	Multi-class	Binary	Multi-class
Learning rate	1e-5	2e-5	1e-3	1e-3
Epochs	4	10	4	10
Batch size	32	32	32	32
Epsilon	1e-8	1e-8	10e-8	10e-8

TABLE IV: Classification results using 2-fold cross validation a) Binary Classification b) Multi-class Classification: Average scores are computed across 5 classes, *INFO, ATK, CVE, MAL and MISC*.

		(a) Binary Classification			(b) Multi-class Classification		
		Precision	Recall	F1-score	Precision	Recall	F1-score
CNN	random	0,80	0,78	0,79	0,65	0,64	0,64
	word2vec	0,86	0,73	0,79	0,73	0,71	0,71
	GloVe	0,81	0,78	0,79	0,72	0,72	0,72
BERT	base	0,84	0,87	0,85	0,80	0,80	0,80
	large	0,84	0,89	0,86	0,80	0,81	0,80
	distil	0,84	0,85	0,85	0,78	0,78	0,77

and validation loss is 4. The epoch graph for BERT-Large SequenceClassifier can be seen in Appendix A.

The BERT-large SequenceClassifier outperforms other models, however BERT-base and BERT Distil SequenceClassifier perform similar in terms of all the performance measures. The main drawback of using BERT-large SequenceClassifier is it requires significantly higher learning time than BERT-base SequenceClassifier i.e. one epoch takes approximately 19 minutes for the former, while it reduces to 6 minutes for the latter one. Whereas, the validation time is almost same for both, and generates the difference of around 50 seconds for one epoch, which is very small.

From the results, it is also evident that the performance of the CNN classifier degrades significantly (recall rate decreases from 0.85 to 0.73) when the amount of training data is reduced from 80% to 50%. Whereas only a slight change in all BERT SequenceClassifier performances are observed because of the transfer learning capabilities. Also, it converges faster at slower learning rate (i.e. 1e-5) in comparison to the CNN classifier (i.e. 1e-3).

Considering the overall performance, we select BERT-base SequenceClassifier for the filtering of irrelevant tweets from our dataset and extracts 33,381 security relevant tweets from the total of 76,047 tweets.

2) Multiclass Classification: After binary classification, the size of original dataset is reduced to 43.9% which indicates that only about half of the collected data is relevant. Unlike other models [6][5], we make a more specific classification of informative tweets while discarding the non-relevant tweets. To do this, we perform multiclass classification on the relevant data extracted from binary classification. We randomly select 1250 samples from this data and manually label each tweet with one of the 5 classes, *INFO, ATK, CVE, MAL and MISC*. Details of the classes are explained in Section III.

TABLE V: Classification results using 5-fold cross validation a) Binary Classification b) Multi-class Classification: Average scores are computed across 5 classes, *INFO, ATK, CVE, MAL and MISC*

		(a) Binary Classification			(b) Multi-class Classification		
		Precision	Recall	F1-score	Precision	Recall	F1-score
CNN	random	0,85	0,81	0,83	0.72	0,70	0,70
	word2vec	0.88	0.85	0.86	0,77	0,76	0,77
	GloVe	0.88	0,79	0,83	0,74	0,74	0,73
BERT	base	0,87	0,88	0,87	0,81	0,82	0,81
	large	0,90	0,89	0,90	0,84	0,84	0,84
	distil	0,86	0,87	0,86	0,82	0,82	0,82

The same models from the previous experiment are chosen for multiclass classification and trained on five classes instead of two. In this case, the optimal number of epochs for SequenceClassifier was found to be 10. An example epoch/loss graph for BERT-Base SequenceClassifier is shown in Figure 6 in Appendix A. The training loss curve flattened at around 7 epochs, but we set it at 10 as it produces slightly better results. The classification results obtained for 5-classes from three variants of BERT embeddings (base, large and Distil) are illustrated in the form of confusion matrices (see Table VI).

According to Table VI, BERT-Large SequenceClassifier is more accurate in classifying the MISC, INFO and CVE classes than other models but struggles a bit with the ATK and MAL classes. This is expected since these are the classes with the most variation in content.

For the CVE class, all classifiers performed almost similar because it has a pattern for each of its samples, e.g. CVE-2020-0761. During prepossessing, numbers were not removed from the data, since they form a significant part of the pattern, thus it made classifiers predict this class correctly. We observed that for all classes except CVE, most false predictions were predicted as MISC.

Performance comparison: The overall classifier performance metrics are calculated by taking weighted average across all classes, shown in Table V(b) and IV(b). For multi-class classification, BERT-large SequenceClassifier performed again best among all. Due to the smaller training time and comparable performance with BERT-large SequenceClassifier, BERT-base SequenceClassifier is chosen to classify the unlabeled relevant tweets. The distribution of relevant samples into *5-classes* is presented in Table VII. This shows that the number of tweets are fairly balanced in all classes except *MISC* category, which holds very few samples (only 2.8%). This signifies that the binary classifier (in previous step) worked well in filtering out irrelevant tweets from the relevant ones. Hence, a few remaining irrelevant tweets are classified as *MISC* during multi-class classification.

	MISC	INFO	ATK	CVE	MAL
MISC	78	17	6	8	19
INFO	9	80	9	1	19
ATK	7	3	112	0	2
CVE	1	0	0	93	0
MAL	6	12	1	3	124

(a) BERT-Base

	MISC	INFO	ATK	CVE	MAL
MISC	91	18	6	6	7
INFO	11	86	12	1	8
ATK	12	4	105	0	3
CVE	0	0	0	94	0
MAL	11	8	5	5	117

(b) BERT-large

	MISC	INFO	ATK	CVE	MAL
MISC	74	20	8	8	18
INFO	13	76	14	1	14
ATK	6	2	111	0	5
CVE	1	0	0	93	0
MAL	9	16	2	2	117

(c) BERT-Distil

TABLE VI: Confusion matrices for multi-class classification using BERT SequenceClassifier

TABLE VII: Relevant tweets distribution

	ATK	INFO	MAL	CVE	MISC	Total
#Tweets	8,038	7,825	8,805	7,781	932	33,381

V. Grouping

In this section, we investigate how meaningful the clusters of tweets generated by CTI-Twitter are for a specific class. We will compare the performance of semantic grouping using k-means and LDA where the majority of results will be presented only for the MAL class due to the page limits of the article. This gives an overview of the selected class by displaying the spread of the data and discovering how many similar groups exists within it.

A. K-Means

As previously explained, the cluster quality highly depends on the k (i.e. number of clusters) parameter. To tune this, we use the elbow method and silhouette score. To implement k-means, the python sklearn library with default settings is used. The best silhouette score is achieved at k = 6 i.e. 0.54. In the elbow graph, a spike is observed at the same value, and this is selected as the optimal k. The plot is shown in Figure 7 in Appendix A.

Clustering is performed on MAL class by translating tweets into vectors using TF-IDF and sentence embeddings. Generally, text embeddings are too large, and can hold hundreds of dimensions, which makes visualization of clusters too hard. To address this issue, principal component analysis (PCA) from python sklearn library is used that reduces the dimension of vectors to 5. This makes the clusters more apparent by addressing the curse of dimensionality. For the visualization of our results, scatterplots are drawn (Figure 2 and Figure 3), where 400 randomly selected samples are plotted in 2d space. The distribution of the samples within each cluster are also shown in Figure 8 and Figure 5.

Th results indicate that TF-IDF clusters are better separated than the SentenceTransformer clusters. However, SentenceTransformer creates more evenly sized clusters. One of the clusters in TF-IDF heavily outnumbers the rest. After manually analyzing the largest cluster, we found that it does not contain a lot of similar data, as there is no specific theme found in it. Furthermore, the SentenceTransformer clusters showed more consistent themes and was found to be the best among the two in our case.

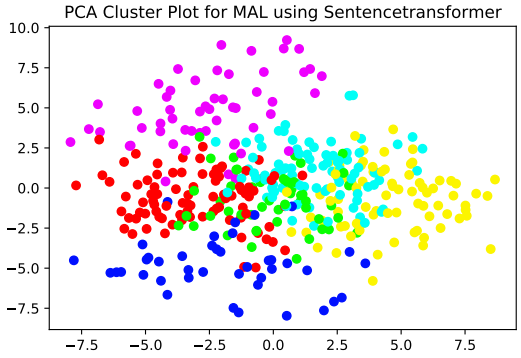

Fig. 2: Grouping of *MAL* tweets using SentenceTransformer embeddings

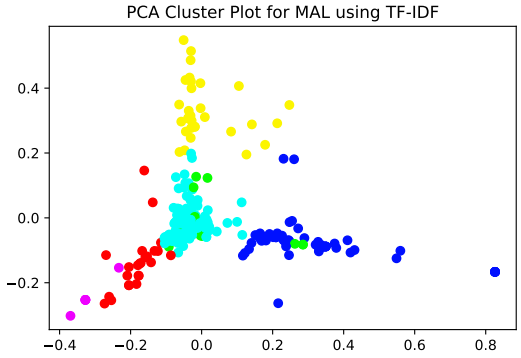

Fig. 3: Grouping of *MAL* tweets using TF-IDF embeddings

B. LDA

To implement LDA, a visualization library named pyLDAvis is used to get an overview of the topics created, trending keywords and to see how much the topics overlap with each other. The number of topics K are set looking at the output from K=2-10. The value of K=6 is selected as it gives the most interpretable and non-overlapping topics. The hyperparameter α, which estimates how many topics are generally in a single document, is set to a low value 0.1. Since tweets are short in length in the Twitter dataset, the number of topics per tweet should be small.

The top 10 words discovered by LDA on the *MAL, ATK and INFO* classes are shown in Table VIII. The keywords under different topics for the *MAL* class mostly

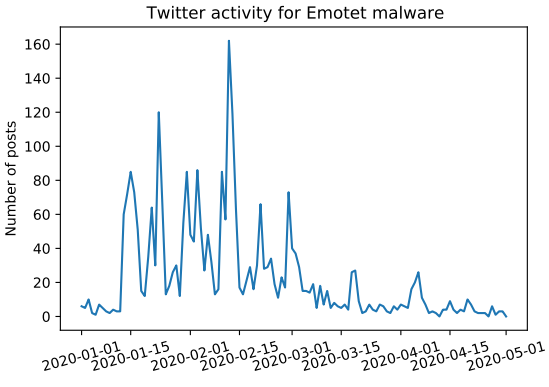

Fig. 4: Activity from Emotet malware

hold different types of malware features, e.g. describing the malware as a Trojan or stating that it has password-stealing functionality. For example, based on MAL topic 3's top words, one can assume that there is malware being distributed through the Android play store.

VI. Feedback and Data Enrichment

In this section, we present further analysis and a few case studies to understand how semantically similar tweets and trending keywords from CTI-Twitter can help investigators and security experts to gain timely and valuable specific threat intelligence.

A. Analyzing trending Keywords from LDA

The grouping of tweets for each class led to the extraction of trending keywords that refers to a certain attack, vulnerability, malware, techniques, tactics etc. Some examples tweets based on trending keywords such as *DoppelPaymer, emotet* etc. from MAL class can be seen in Table IX. Some text is kept hidden due to Twitter's privacy policy.

Furthermore, these keywords can be used to construct a targeted query that is sent back into our model to collect more information about the threat, as they are malware names and will mainly return tweets discussing the specific malware. The streaming API often results into a large number of tweets containing the current trending keywords, whereas the tweets with past trending keywords appears less. In this situation, historical search can yield better results for keywords not tweeted about in the current month.

'Emotet' on Twitter: *Emotet* is a word that appeared frequently in our dataset (can be seen in Table IX on top in topic 5 under MAL class). We performed historical search with the query "*emotet malware*" from January 1st to April 30th and plotted the time-series pattern to see attack trends and 'Emotet' activity over the period. From the Figure 4, the first large spike is observed around mid-January.

To confirm this, we did a Google search and found that on January 22. 2020, the US national cyber awareness system made a blogpost warning people of increased activity from the malware [11]. Also, the behaviour of the time-series shows that the frequency of tweets for the malware decreased after the 1st of March, thus, indicating less activity.

B. A Case Study: Covidlock

This case study is presented to illustrate the usefulness of the proposed CTI-Twitter in discovering a set of IoCs and identification of attack *techniques, tactics and procedures*(TTPs) with verification from authorized sources.

During topic modeling using the pyLDAvis tool, the 'Covidlock' keyword was discovered under topic 2, though not in top-10 keywords, from the MAL class. Using our system, we extract the real-time tweets related to the keyword, and summarize the following information:

1) The term appeared in the dataset starting March 13^{th}, 2020.
2) This is a ransomware targeting Android systems.
3) Ransom demanded to be paid in bitcoin.
4) Disguised as a coronavirus tracking app.
5) Locks down the phone after app download.
6) A firm released the decryption key: 4865083501.
7) Malicious domain found by the DomainTools team.

Thereafter, we performed historical search on 'covidlock' between 10-23 March, and collected more relevant tweets. The summary of 'covidlock' intelligence from this stage is given below:

1) Demands 100$ in bitcoin within 48 hours.
2) Changes screen lock password of infected device while asking for a password
3) Does not actually encrypt/decrypt any files, this is purely a screen-lock attack
4) The device should have a password to prevent this attack

We verified the intelligence with the DomainTools research [15] that confirms that most of the information is correct, including the decryption key. However, the company also provided additional information which was not found in our collected tweets. For instance, malicious domain names and the origin of the attacker. Furthermore, an in-depth analysis of the malware code was conducted by them [15], which is information one generally can not find in a single tweet due to its length restrictions.

According to Google, the earliest search hit for the term covidlock was on the 12^{th} March, 2020, which is one day before it was detected by our proposed model. This indicates the effectiveness and timeliness of the proposed CTI-Twitter methodology in gaining attack-specific knowledge (TTPs and IoCs) using trending keywords generated from semantic grouping.

TABLE VIII: List of Top-10 keywords for each topic under *MAL, ATK and CVE* class

Class	Topic	Top-10 keywords
MAL	0	coronavirus, data, passwords, app, maps, beware, pcs, doppelpaymer, phishing, infecting
	1	windows, shellcode, linux, group, computer, known, bypass, based, dns, sample
	2	trojan, sites, bitcoin, google, fears, macos, disguised, shlayer, fa, use
	3	play, apps, store, new, industrial, files, android, used, control, latest
	4	facebook, remote, ransom, driver, strain, uses, network, cookiethief, accounts, hidden
	5	**emotet**, metasploit, mac, iot, virus, networks, apple, tool, ios, hacking
ATK	0	phishing, north, world, targeting, warns, tesla, compromised, risk, covid, businesses
	1	firm, chubb, insurance, targeted, insurer, cybercrime, maze, finastra, company, fintech
	2	city, council, year, chinese, recent, redcar, cyberattacks, racine, defense, health
	3	breach, hacked, russia, georgia, databreach, data, power, travelex, israel, reports
	4	gas, pipeline, ryuk, healthcare, construction, natural, infection, computers, industry, days
	5	report, iran, state, campaign, internet, toll, group, computer, iranian, wannacry
	6	bitcoin, court, hospital, university, linked, high, freeze, firms, law, uk
INFO	0	android, cyberattack, cloud, infosec, mobile, hack, cybersecurity, targeted, backups, vulnerabilities
	1	average, google, billion, ransom, cost, million, maze, rise, health, industrial
	2	phishing, businesses, microsoft, scams, scam, latest, internet, bitcoin, human, breaches
	3	coronavirus, hackers, using, spread, government, pandemic, iot, maps, people, advantage
	4	vulnerability, windows, day, exploit, nce, vigil, software, code, critical, remote

TABLE IX: Samples tweets from MAL class grouped under *topic 0 and 5*

	Tweets
Topic 0	1. U**** FTCODE R****ware Steals **Passwords**
	2. **DoppelPaymer** Ra****** **Data** f**m Supp***** to SpaceX, T****
Topic 5	1. **Emotet** malware infects n*** insecure W*F* **net***
	2. **IoT** m****e Mirai h** s*** increase in up***es as new variant emerge

VII. Conclusion and Future Work

This article discovered that Twitter holds a lot of real-time CTI that can give insights into current attacks, trending malware and vulnerabilities. Yet, data collected from Twitter consists of many duplicates and irrelevant texts, thus an effective way of filtering without losing relevant information is required. CTI-Twitter provides an effective approach for filtering tweets using a BERT Sequence Classifier, which reached an average precision of 88% on the binary set, and 82% on the multiclass set. When grouping posts together using LDA, it gave us meaningful groups on our data, and trending keywords could be identified. Lastly, data enrichment was found to enhance the collected CTI using identified keywords. With these, we constructed threat activity timelines and better search queries to gather more information. For future work, CTI-Twitter could be improved by i) implementing a ranking model (using timestamp and frequency attributes) for effective prioritization of tweets and ii) test more grouping approaches that work well on short texts, such as Biterm topic modeling.

Acknowledgements

The findings presented in this article are based on the Master's thesis work [7] of one of the authors. The research work is funded by the Department of Information Security and Communication Technology (IIK), Norwegian University of Science and Technology, Gjovik, Norway.

References

[1] Al-khateeb, S. Deviance in social media and social cyber forensics : Uncovering hidden relations using open source information (osinf), 2019.

[2] Blei, D., Ng, A., and Jordan, M. Latent dirichlet allocation. *Journal of Machine Learning Research 3* (05 2003), 993–1022.

[3] Deliu, I., Leichter, C., and Franke, K. Extracting cyber threat intelligence from hacker forums: Support vector machines versus convolutional neural networks. In *2017 IEEE International Conference on Big Data (Big Data)* (12 2017), pp. 3648–3656.

[4] Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR abs/1810.04805* (2018).

[5] Dionisio, N., Alves, F., Ferreira, P. M., and Bessani, A. Cyberthreat detection from twitter using deep neural networks. *CoRR abs/1904.01127* (2019).

[6] Le, B. D., Wang, G., Nasim, M., and Babar, M. A. Gathering cyber threat intelligence from twitter using novelty classification. *CoRR abs/1907.01755* (2019).

[7] Linn-Mari, K. Gathering Open Source Threat Intelligence from Twitter: An Integrated Supervised and Unsupervised Learning Approach. Master's thesis, Norwegian University of Science and Technology, Norway, 2020.

[8] Ma, G. Tweets classification with bert in the field of disaster management.

[9] Mackey, T., Kalyanam, J., Klugman, J., Kuzmenko, E., and Gupta, R. Solution to detect, classify, and report illicit online marketing and sales of controlled substances via twitter: Using machine learning and web forensics to combat digital opioid access. *Journal Of Medical Internet Research 20*, 4 (2018).

[10] Mittal, S., Das, P. K., Mulwad, V., Joshi, A., and Finin, T. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities, 2016.

[11] NCASystem. Increased emotet malware activity.

[12] Park, S. J., Lim, Y. S., and Park, H. W. Comparing twitter and youtube networks in information diffusion: The case of the "occupy wall street" movement. *Technological Forecasting & Social Change 95* (2015), 208–217.

[13] Ramos, J. Using tf-idf to determine word relevance in document queries.

[14] Reimers, N., and Gurevych, I. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on*

Natural Language Processing (EMNLP-IJCNLP) (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 3982–3992.
[15] Saleh, T. Covidlock update: Deeper analysis, 03 2020.
[16] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.
[17] Sapienza, A., Ernala, S. K., Bessi, A., Lerman, K., and Ferrara, E. Discover: Mining online chatter for emerging cyber threats. pp. 983–990.
[18] Zhu, J., Tian, Z., and Kübler, S. Identifying offensive tweets using bert and svms, 04 2019.

Appendix A

Performance Metrics

The performance measures are calculated using four parameters i) TP (true positive): the number of relevant samples predicted as relevant. ii) FP (false positive): the number of irrelevant samples predicted as relevant. iii) FN (false negative): the number of relevant samples predicted as irrelevant. and iv) TN (True negative): the number of irrelevant samples predicted as irrelevant.

The measures are described below:

- **Precision** Fraction of the actual relevant samples that are predicted as relevant i.e.

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP})$$

- **Recall** It is defined as the correctly predicted relevant samples divided by all relevant samples i.e.

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN})$$

- **F1-score** is measured as the weighted average of the precision and recall scores.

$$\text{F1} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

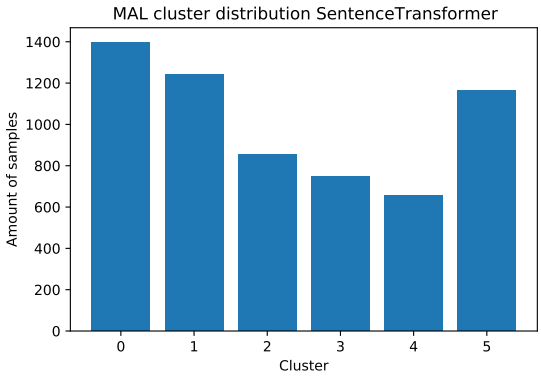

Fig. 5: Sentencetransformer clusters

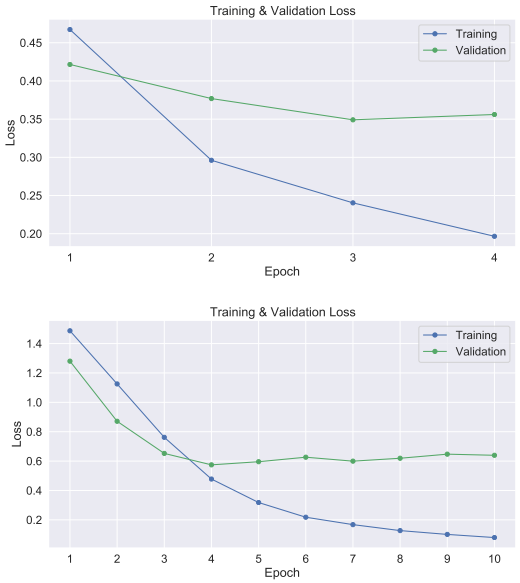

Fig. 6: Loss variation using BERT-Large SequenceClassifier for binary (top) and multiclass (bottom) classification

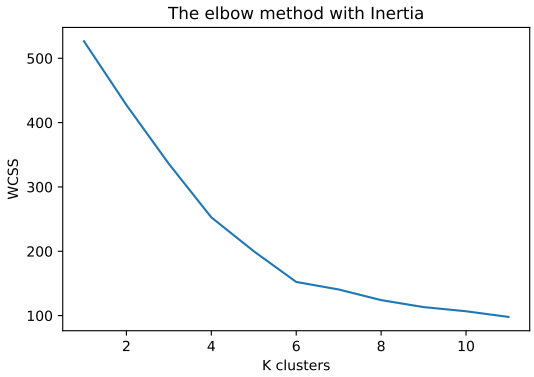

Fig. 7: MAL Class elbow

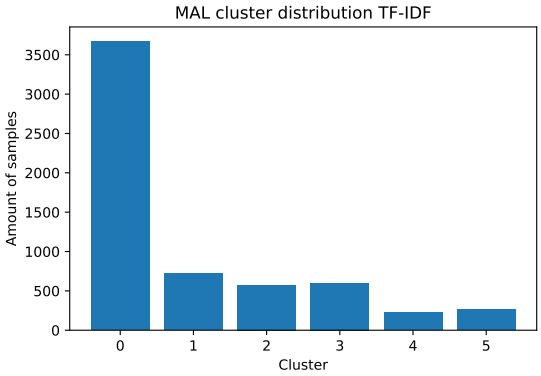

Fig. 8: TF-IDF clusters