A Novel Approach for Detection and Ranking of Trendy and Emerging Cyber Threat Events in Twitter Streams

Avishek Bose, Vahid Behzadan, Carlos Aguirre, William H. Hsu
abose@ksu.edu, behzadan@ksu.edu, caguirre97@ksu.edu, bhsu@ksu.edu
Department of Computer Science, Kansas State University, Manhattan, Kansas, 66506, USA

***Abstract*—We present a new machine learning and text information extraction approach to detection of cyber threat events in Twitter that are *novel* (previously non-extant) and *developing* (marked by significance with respect to similarity with a previously detected event). While some existing approaches to event detection measure novelty and trendiness, typically as independent criteria and occasionally as a holistic measure, this work focuses on detecting both novel and developing events using an unsupervised machine learning approach. Furthermore, our proposed approach enables the ranking of cyber threat events based on an importance score by extracting the tweet terms that are characterized as named entities, keywords, or both. We also impute influence to users in order to assign a weighted score to noun phrases in proportion to user influence and the corresponding event scores for named entities and keywords. To evaluate the performance of our proposed approach, we measure the efficiency and detection error rate for events over a specified time interval, relative to human annotator ground truth.**

***Index Terms*—novelty detection, emerging topics, event detection, named entity recognition, threat intelligence, user influence, tweet analysis**

I. Introduction

This paper presents a new methodology for recognizing potential cyber threats using passive filtering and ranking in social text streams, particularly Twitter streams. *Passive* monitoring here refers to collecting intelligence and solutions of different cyber threats from different platforms using only text corpora and lists of named entities or keywords (e.g., gazetteers) rather than direct background knowledge of threats. Twitter is examined as a high-bandwidth platform where both actors from both sides of cyberdefense, such as attackers and security professionals, post cybersecurity-related messages [19]. The overall goal of this work is to analyze these messages collectively to attain actionable insights and collect intelligence on emergent cyber threat events. Detecting events from social media includes a) **novel** event detection, including first stories or tweets about previously non-extant topics; and *developing* events (especially for bursty topics, but also for non-bursty topics for which volume and aggregate important build up gradually). In this work, we treat novelty of events from the developing nature of events (emergence) as **orthogonal** properties. This allows novel events that have not yet attained trending status or viral propagation to be tracked, while still incorporating traditional trend detection methods.

ASONAM '19, August 27-30, 2019, Vancouver, BC, Canada
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6868-1/19/08?/$15.00
http://dx.doi.org/10.1145/3341161.3344379

Recent research includes some work on detecting both novel and developing events in Twitter streams (e.g., [1][10]), especially where emergence is defined as trending. However, only a few studies have further focused on detecting cyber threat events in Twitter streams. Furthermore, we propose an approach to the ranking of events with regards to their significance. While such a ranking generally depends on both the application domain and user objectives, the relative importance to a general community of interest, such as the cybersecurity community, can be imputed based on pervasiveness, spread rate, and novelty. In this study, we also rank the two types of events based on the order of their corresponding importance score to show how much a particular event is important compared to proximate events within a user-specified time range of a reference tweet. In contrast with large document corpora, analyzing short documents such as tweets presents some specific semantic challenges towards extracting terms, relationships, patterns, and actionable insights in general. For example, terms mentioned in a short tweet lack context, and there is less co-occurrence data in the entire corpus on which to base expressible relations between named entities or terms.

Our system takes as input a user-specified maximum interval of detection for related cyber threat events, within an original tweet that is deemed relevant. The full text of this tweet, or quoted part of a retweet, is captured. Social network parameters such as indegree (number of followers) are calculated and normalized by range. The text bodies of tweets are vectorized using term frequency-inverse document frequency (TFIDF) [17], the resulting TFIDF vectors are clustered using the *DBSCAN* [16] density-based clustering algorithm, noise points are discarded, and the concatenated text contents of each cluster are ranked using the *TextRank* algorithm [14], to obtain representative keywords and named entities that represent **potential** events. We then identify different scenarios: a) novel and developing story; b) novel story only; c) developing story only; d) not an event based on heuristics that are described in Section IV. Additionally, we also calculate an importance score for each event based on the heuristics presented in Section IV. Finally, we tag each event according to their descriptive features and provide a rank based on their importance scores.

Key **novel contributions** of this work are as follows:

1) We detect both trendy and novel types of events related

to cybersecurity from Twitter streams.

2) We provide a method for the ranking of potential cyber threat events according to their importance score based on keywords, as well as their named entity confidence and user influence scores.
3) The proposed method can be tuned to capture important cybersecurity events based on user-specified parameters.

II. Related Approaches

This section briefly summarizes key methodologies for cyber threat detection from text corpora, particularly social media.

Dabiri et al.[3] analyzed traffic related tweets for detecting traffic event by applying deep learning models, including convolutional and recurrent neural networks incorporating a `word2vec`-based word embedding layer to represent terms. This approach performs well but is domain-dependent and highly costly in terms of manual annotation for high-throughput sources of training data such as Twitter. In contrast, *TwitInfo* [2] incorporates a new streaming algorithm that automatically discovers peaks of event-related tweets and labels them from the tweets texts. This approach, however, focuses only burstiness of tweets and ignores both user influence and novelty with respect to developing events.

Rupinder et al. [6] also proposed a framework based on deep learning for extracting cyber threat and security-related insights from Twitter, categorizing three types of threats (examples of which are Distributed Denial of Service (DDoS) attacks, data breaches, and account hijacking). From text documents, events are extracted using a) target domain generation; b) dynamically-typed query expansion; and c) event extraction. This approach employs both syntactic and semantic analysis using dependency tree graphs and convolutional kernel, but is highly computationally intensive due to the cost of autoencoder training.

Sceller et al. [9] uses unsupervised learning to detect and categorize cybersecurity events by analyzing cybersecurity-related Twitter posts based on a set of seed keywords specified for each level taxonomy. This algorithm is prone to false negatives because it may not detect potential cyber threat events as events in the first place.

Mittal et al. [4] propose a method for processing threat-related tweets using the Security Vulnerability Concept Extractor (SVCE) which generates tags about cybersecurity threat or vulnerabilities such as means of an attack, consequences of an attack, affected software, hardware, and vendors. This approach does not generalize to the user communities as it is personalized for individual users' system profiles.

Edouard [5] propose a framework that utilizes Named Entity Recognition (NER) and ontology reasoning (using *DBPedia*), along with classification learning approaches such as Naive Bayes, SVM, and a deep neural network (Long Short Term Memory / Recurrent Neural Network, aka LSTM-RNN), for category tag imputation. The graph algorithm *PageRank* is used to rank candidate items for information retrieval.

The approach of Lee et al. [7] focuses on community communication and influence to detect cyber threats by grouping highly contributing Twitter users and scores them as an expert community to get information to be explored and then to be efficiently exploited. This framework incorporates four components: a) an interface to the Twitter social media platform; b) a flexible machine learning system interface for document categorization; c) a mixture-of-experts weighting and extraction scheme; d) a new topic detector. This framework is highly dependent on expertise and data quality.

A method by Sapienza et al. [8] considers various web data sources to generate indication of warnings for detecting upcoming potential cyber threats. While potentially extensible to named entites discoverable by set expansion, this approach is focused on detecting "novel words" and does not yet incorporate a full contextualized topic model, feature weighting model, or method of user influence.

Finally, the work of Alan et al. [11] is based on a supervised learning approach to train an extractor for extracting new categories of cybersecurity events by seeding a small number of positive event example over a significant amount of unlabeled data. As with previous approaches, it does not yet incorporate full NER nor allow for entity set expansion.

III. Background

This section presents a brief review of the key technologies that are adopted in our proposed framework for threat event detection in tweets.

A. Named Entity Recognition (NER)

In general, NER is an information extraction task aimed at locating and classifying the names of specific entities such as persons, organizations and locations, based on analysis of text units such as n-grams and noun phrases. Generic entities such as numerical quantities are sometimes also included. In our analysis, NER is used to discover the names of entities in reported cyber threats. Key objectives of using machine learning to improve NER are: a) set expansion to broaden the set of cyber threats based on synonymy and other relationships that can be inferred by text pattern analysis; b) feature weighting for relevance or salience; c) relationships that are discoverable from data; d) confidence scoring.

B. TextRank

The TextRank algorithm [14] is an extended version of Google PageRank [15] algorithm that aims to determine keywords by generating a word graph from a given text document unlike determining high ranked webpages that is done by the PageRank algorithm. The TextRank score calculates importance of a word from given a text that is identical to PageRank score works for webpages. The importance however associated with a vertex is determined based on the votes that are cast for it, and the vertices' score casting these votes.

C. TFIDF

TFIDF [17] is a information retrieval method used in various purpose such as word co-occurence based document vectorization, word ranking, document similarity calculation, etc. In information retrieval, TF (term frequency) refers to term frequency of a particular word in a document, while IDF (inverse document freqency) refers to inverse document frequency of a word in the whole corpus of documents.

D. DBSCAN

DBSCAN [16] is a density-based clustering approach works by enforcing a minimum number of data points (*MinPts*) inside a specified-radius neighborhood (*Eps*) of a data point to make a *density-reachable* cluster; this process continues until no points on the frontier are density-reachable, then restarts with a new initial point.

IV. Proposed Method

Analyzing Twitter texts for getting valuable insights has always been an issue because of its unstructured way of writing and the length of tweets. In this section we go through some steps described in the subsections below.

A. Tweet Collection and Early Annotation

In this analysis we used Twitter data collected through four days from 6^{th} September to 9^{th} September in 2018 and a small portion of data collected in 30^{th} and 31^{th} August, 2018 that is stored in MongoDB database. As our main focus is to getting insights of cybersecurity-related events and rank their scores, we crawled Twitter data using the Twitter API based on some security related keywords. Without applying security related keywords, the crawled Twitter data would be generalized to all domains and thus the result would be biased to detecting general kind of events. This datasets was manually annotated whether the tweets are relevant to cyber security or not by taking help from four annotators for our earlier work [19]. Although we used security related keywords for crawling the Twitter data, many of those tweets are irrelevant or promotional. That is why the annotation plays a crucial role here and the resulting dataset of this process are available at [12]. In this study, we initially have 21368 tweets and working with the annotated data, we found 11111 tweets are related to cyber security. We apply our algorithm on those cyber security related tweets and the whole tweets' datasets individually. We took the full text of each tweet if the tweet is not quoted or retweeted from any other users. If any tweet is retweeted or quoted from any other user, we take the original retweeted or quoted tweet full text. Then, we let an user to give a numeric value input as the number of time intervals based on tweet occurrence. This process divides the whole time period of tweet occurrence into some equal time chunks based on the number that we are taking from the user. Thus, for each time interval a number of tweets are aggregated into a chunk based on their corresponding occurring time.

B. Tweet Pre-processing and Cleaning

As we stated earlier that tweet text is very unstructured that contains a lot of misspelled words and sometime the text is not a complete sentence. That is why we apply a tool named SymSpell [18] to correct the misspelled words. Then we take only the characters from the tweet text that are alphanumeric and remove all punctuation characters. Then we remove all the stopwords from the text because these are so frequently occurred over the whole data set that may reduce the analysis performance. After that we consider cleaning the tweets' texts in both cases either by applying stemming or without stemming. We removed all the words or tokens which lengths are only one.

C. Influential Twitter user Impact

Influential users tweets are valuable for detecting important cyber security related events. That is why we keep records of each Twitter user with their number of followers corresponding to their posted cyber security related tweets. As the number of followers represents is directly proportional to the influence of the user, their used words in cyber security related tweets are also important. So, for each time interval, we normalize the values of follower numbers for each user using Min-Max normalization. This normalization process normalizes each value between 0 and 1. We then assign the normalized value of users' follower number to each of their used noun phrases in tweets. Here, we used python nltk library to extract noun phrases from tweets. If similar words are used by several tweet user, we keep the highest normalized value of an user for each word used in a tweet. Now, for each time interval, for all tweets, each noun phrase has a corresponding value that represents its weight inherited by its user. This value will also be used to calculate event score.

D. Determining Algorithm Design Architecture

In this study, we apply a very popular word vectorization method in NLP domain named tfIdf [17] that is based on word co-occurrence in documents to make word vector for each tweet from the data set. After doing process mentioned above for all tweets in a time interval, it generates a tfIdf matrix. We found this method gives a better performance compared to the word semantic relation based approaches that are discussed later in Section VI. Then, we apply the DBSCAN [16] density based clustering algorithm using the aforementioned tfIdf matrix to find cluster of similar meaning tweets. These clusters can represent the potential events. However, we did not apply the K-means clustering algorithm because we did want to limit the number of events found in our analysis, for each time interval. For this analysis, we ignore the noise points generated by applying DBSCAN because in our observation over the data set we found that the noise points are conveying a very little impact to find cyber security related events.

E. Event Detection Heuristics and Scoring

Firsly, we aggregate all tweet texts in a cluster into a single text. Then we simulate named entity and keyword identification process on the aggregated text by applying TextRazor [13] online Named Entity recognition API and TextRank [14] algorithm from Gensim library respectively. Additionally, the TextRazor provides a Confidence score for each named entity and TextRank from Gensim [20] also provides a score for each keyword based on word graph mentioned earlier in Section IV. We apply two different set rules to determine the type of the events and and their corresponding score that will be used to rank the cyber security related events. To formulate our idea into implementation, we produce four different sets of token set mentioned below.

1) *commonSet*-refers the set of words that are common in both named entity and keyword. Additionally, we also take some higher scored named entities and keywords from two sets *namedEntitySet* and *keywordSet* mentioned later respectively to add the tokens to the *commonSet*.
2) *keywordSet*-refers the set of words that only appear in the keyword set and not common with the named entity set.
3) *namedEntitySet*-refers the set of words that only appear in the named entity set and not common with the keyword set.
4) *unionSet*-this set keeps all named entities and keywords.

The Figure 1 clearly depicts the graphical illustration of four different token sets mentioned above. Here *commonSet*, *keywordSet*, *namedEntitySet* and *unionSet* are represented by $k \cup (K \cap N) \cup n$, *K-N*, *N-K* and $K \cup (K \cap N) \cup N$ respectively. Here, *k* and *n* are the set of highly scored keywords and named entities and can be represented as $k \in K$ and $n \in N$ respectively.

1) Determine Event Novelty: We store all the tokens of the set *namedEntitySet* and *commonSet* into another set of tokens named as *noveltyCheckerSet* for all clusters generated for all time intervals. We are storing all these tokens because we are checking the similarity of the tokens from the set of a subsequently generated cluster to the stored tokens' set *noveltyCheckerSet* to determine whether the newly generated cluster has some novelty or not based on a cosine similarity threshold value determined empirically.

2) Determining Trendiness: If the similarity score reaches the defined threshold *cosineThresh*, we determine the working cluster is just trendy except the very first cluster because this cluster would not find any different set to compare the similarity. We also take a user defined threshold of number of tweets *tweetThresh* to determine getting a trendy event. Thus, if the the number of tweets does not reach to the value mentioned by the user, it will be an unnoticeable event. However, if tweets from a cluster satisfies the cosine similarity threshold *cosineThresh* as well as number of tweets threshold *tweetThresh*, still it may not represent a noticeable event because it may be a spamming of banal topic. So, we apply a different heuristic if the length of the *commonSet* set is greater than the one fifth(0.20) times of the *namedEntitySet* set, only then the cluster will be counted as trendy. We are checking this because a big cluster of tweets' texts will have so many named entities that will mean a variety of topics but a single cluster should be biased towards a single topic described in it.

3) Determining Novelty: Now, if the cosine similarity of stored token set of tokens *noveltyCheckerSet* and working cluster token set is less than the threshold value *cosineThresh*, the working cluster could be a potential novel event. However, we set a threshold value minimum three tweets to be in the cluster to refer it as an event. Now, if the number of tweets is greater or equal than the user defined threshold value of trendiness *tweetThresh*, the cluster is addressed as "Novel and Trendy" but if the number of tweets in the cluster is less than the number of user defined threshold value, it is addressed as "Just Novel". That is how we determine a type of a generated cluster of tweets as an event type.

4) Event Score Calculation Process: As we mentioned earlier the ranking of cyber threat related events is also an important task, we are motivated to calculate score of each event if an event finds a event type based on our empirically defined heuristics mentioned above. We calculate scores for each of defined events individually by applying different heuristics. As the confidence score of a named entity in the TextRazor and the score of a keyword in the TextRank algorithm are different in scale, we apply sigmoid function to normalize each scores of a named entity and keyword respectively. As every token is stored in a dictionary for each generated cluster, we update the score of each token if it considered both named entity and keyword by adding two scores after normalizing by the sigmoid function.

5) Score Calculation for Trendy Event: Now, for a "Just Trendy" event firstly we calculate the entity score of the event by adding the scores of each token included in the *commonSet* and then multiplying the total added value with the value of total number of tweets that makes an event as trendy. Secondly, we added the value of each noun phrases corresponding to the event's tweets where the noun phrases are inherited from the value of influential users' followers. This score are then added to the initially calculated entity score mentioned above for the aggregated tweets' texts to get the total score. We calculate the entity score like this mentioned above because, this will assign a higher score to a trendy event either if tweets in a same topic appear so many times in a cluster or number of tokens in the *commonSet* is higher. This heuristic assumes that even if the event topic does not appear so many times in corresponding tweets compared to the other highly appeared event topics, because of the number of common tokens in both name entity and keywords, the heuristic give importance to those tokens as important tokens.

6) Score Calculation for Trendy and Novel Event: Then for a "Novel and Trendy" event, we added the values of all the tokens of the set being generated by the union operation of *keywordSet* and *commonSet*. Afterwards, we multiply the added value with the value of total number of tweets that represent the event to calculate the entity score. Then we again added the value of each noun phrases corresponding to the event's tweets where the noun phrases are inherited from the value of influential users' followers. This score are then added to the entity score like discussed earlier in the above paragraph to get the total score. We calculate the entity score like this because this event is already proved to be a novel event and we need to consider whether it has tokens common in both named entity and keyword set to check the main topic discussed about in this event. Additionally, we also need to consider the keywords appeared in this event to check which topics are also mentioned in the tweet texts of the novel event. That means a novel and trendy event will get a very higher score compared to the other events.

7) Score Calculation for Just Novel Event: Now, for the "First Story" event, we consider to keep a set of tokens generated by differentiating *keywordSet* from *unionSet* and then doing union operation with the *commonSet*. This resulting set stores all the named entities with the keywords which have very high score. Then, we add all the values of the corresponding tokens in the resulting set and multiply the added value with the user defined threshold value *tweetThresh* for being an event as trendy to get the entity score. Then we repeat the procedure of adding the value of noun phrases to the entity score of the working event. We are calculating the entity score like this, because if the event is just novel, it will not appear in so many tweets and that is how it may loose its importance. So, by means of giving importance to this kind of events, we are multiplying the total score of resulting set of tokens by the value of *tweetThresh*. Thus, it can get at least as importance as any trendy event can acquire. We choose the aforementioned resulting set because, we need to consider the novelty of an event that is based on the the confidence score of the named entity and some high ranked keywords. There is no mean to consider the whole keyword set right now because in this case, it seems useless to other trendy topics discussed except the common ones with named entity set.

8) Ranking Scores of Events: Our proposed approach repeats the above mentioned condition checking for each cluster whether to determine as an event or not and score calculation for each cluster that is only considered as an event for each time interval. Finally, we rank each event by ordering their total score for each time interval. The flow chart of our proposed approach is depicted in Figure 2.

F. Annotation Approach

To evaluate the performance of our proposed approach, we compared the results of our proposed method with a manually annotated list of events. A subset of 301 tweets collected in sequence in the window starting at 2018-08-30 23:00:08 CST to 2018-09-02 10:50:19 CST was manually annotated according to i. impact, ii. tweet count, and iii. worldwide effect to be considered as an event. We also consider three categories whether they are i. first story and novel, ii. trending or developing, and iii. novel and trending. For validation, we check the ratio of correctly detected events in that window to the total number of relevant events, and the ratio of correctly detected events to the total number of detected events.

V. Experimental Results

A. Simulation

For our analysis, we use scikit learn [21] to calculate tfIdf matrix [17], to apply DBSCAN [16] algorithm and to use cosine similarity. The parameters assignments for making the tfIdf matrix are *max_df* = 0.90, *max_features* = 200000, *min_df* = 0.01 and *ngram_range* = (1,1). Then, parameter assignments for DBSCAN algorithm are *eps* = 1 and *min_samples* = 3. Again, we use the cosine similarity threshold as 0.5 for similarity checking for trendiness. Table I shows

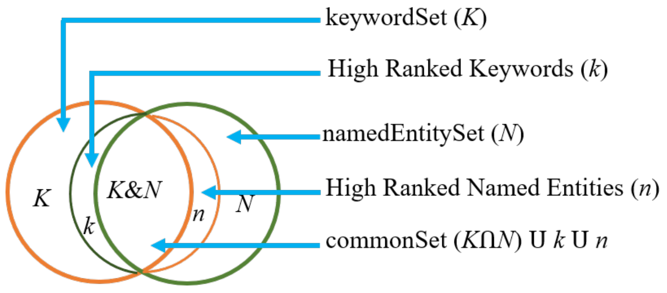

Fig. 1. Graphical representation of *commonSet*, *keywordSet* and *namedEntitySet*

TABLE I
SUMMERY RESULT OF FIVE TIME INTERVALS; NT:NUMBER OF TWEETS; JT: JUST TRENDY; TN: TRENDY AND NOVEL; FS: FIRST STORY; TE: TOTAL NUMBER OF EVENTS

Interval	NT	JT	TN	FS	TE
1	145	0	1	14	15
2	314	0	0	50	50
3	812	1	7	37	45
4	1239	0	9	18	27
5	297	4	0	5	11

the result of five time intervals collectively from 2018-08-30 23:00:08 to 2018-09-02 10:50:19.200000, from 2018-09-02 10:50:19.200000 to 2018-09-04 22:40:30.400000, from 2018-09-04 22:40:30.400000 to 2018-09-07 10:30:41.600000, from 2018-09-07 10:30:41.600000 to 2018-09-09 22:20:52.800000 and from 2018-09-09 22:20:52.800000 to 2018-09-12 10:11:04 by Interval 1, 2, 3, 4 and 5 respectively. We keep only detected True Positive events and represent in Table I. From Table I we can see for the first interval (2018-08-30 23:00:08 to 2018-09-02 10:50:19.200000) we have total 145 tweets and total 15 events. Moreover, out of 15 events we got no "Trendy Event", 1 "Novel and Trendy Event" and 14 "Novel Event". This description will be continued for rest of the intervals similarly. In Table II we show the extracted keywords for each event for the first time interval. The keywords mentioned in the Table II are used to detect cyber security related events for the first time interval. For better representation, We show only the plot of the 2^{nd} time interval in Fig. 3 because of the paper space limitation. This figure depicts the found events in x axis, the amount of tweets on left side of figure 3 in y axis and the event score on the right side of figure 3 in y axis. The red vertical bar represents number of tweets and the blue vertical bar represents event score for each detected event.

B. Annotation-Based Validation

1) Design Selection Approach: There are few decision we had to make to formulate the design architecture of this algorithm. Firstly, we thought to analyze the semantic relation between the words of each tweet text to get the insights of cyber security events. That is why we previously applied doc2Vec [22] which is an extended application of word2vec [24] for getting similar meaning tweets to find events from the data sets but we could not get any satisfactory result because we found that shallow neural network model text domain tools like doc2Vec [22] works based on word vector

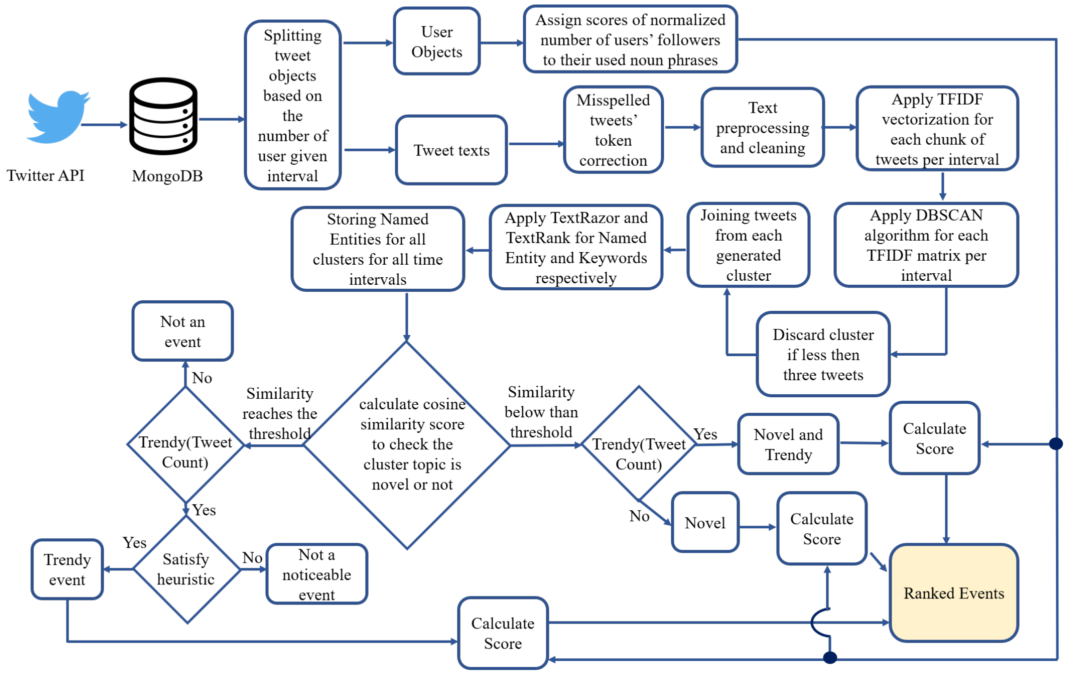
Twitter API
MongoDB
Splitting tweet objects based on the number of user given interval
User Objects
Assign scores of normalized number of users' followers to their used noun phrases
Tweet texts
Misspelled tweets' token correction
Text preprocessing and cleaning
Apply TFIDF vectorization for each chunk of tweets per interval
Apply DBSCAN algorithm for each TFIDF matrix per interval
Joining tweets from each generated cluster
Apply TextRazor and TextRank for Named Entity and Keywords respectively
Storing Named Entities for all clusters for all time intervals
Discard cluster if less then three tweets
calculate cosine similarity score to check the cluster topic is novel or not
Similarity reaches the threshold
Similarity below than threshold
Trendy(Tweet Count)
Not an event
No
Yes
Satisfy heuristic
Not a noticeable event
Trendy event
Novel and Trendy
Novel
Calculate Score
Ranked Events

Fig. 2. Flowchart of the proposed approach

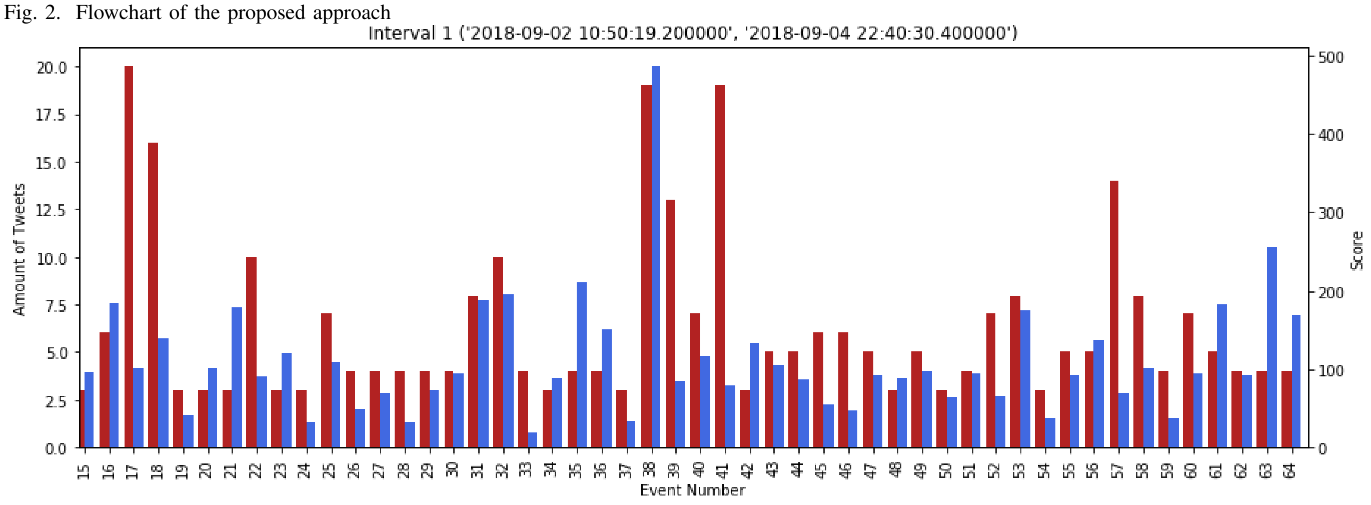

Fig. 3. Event plot of the second time interval proposed approach

TABLE II
SUMMERY RESULT OF TIME INTERVAL 1('2018-08-30 23:00:08', '2018-09-02 10:50:19.200000')

Event Number	Keywords
0	'security', 'android (operating system)', 'android', 'wi-fi', 'privacy'
1	'microsoft', 'disclosed', 'twitter', 'windows', 'microsoft windows', 'hacker discloses'
2	'website', 'catalonia', 'spain', 'banking', 'bank', 'inf'
3	'based', 'huff', 'buffer overflow'
4	'vulnerability (computing)', 'security', 'repository', 'critical vulnerability', 'apache', 'inf'
5	'vulnerability', 'resource consumption', 'prior', 'resource', 'rsa', 'bleach'
6	'task','windows','patch,'scheduler'
7	'security', 'android (operating system)', 'android', 'data', 'privacy', 'tracking'
8	'cracking ransom', 'coin', 'free', 'ransom', 'cybersex', 'net'
9	'vulnerability (computing)', 'patch', 'spyware', 'phishing', 'inf sec cube security', 'patched', 'malware'
10	'security', 'website'
11	'plus', 'pump'
12	'version', 'web server', 'debugger', 'skype', 'update', 'denial service', 'exploit (computer security)'
13	'cisco systems', 'service', 'cisco'
14	security', 'photo','service'

TABLE III
SUMMERY RESULT OF TIME INTERVAL 1('2018-08-30 23:00:08', '2018-09-02 10:50:19.200000');EN: EVENT NUMBER;EL: EVENT LINK; TC:TWEETS COUNT
NESR:NORMALIZED EVENT SCORE RANK;ET: EVENT TYPE; AER: ANNOTATOR EVENT RANKING; DBR: DIFFERENCE BETWEEN RANKINGS; FS: FIRST STORY (NOVEL); FST: FIRST STORY AND TRENDY (DEVELOPING)

EN	TC	Event Score	NESR	ET	EL	AER	DBR
0	5	167.6084	5	FS	link1	4	1
1	7	211.033	3	FS	Link2	2	1
2	21	190.5950	4	FS	Link3	3	1
3	3	55.3226	10	FS	NA	13	3
4	12	110.6048	9	FS	Link4	7	2
5	4	130.4169	8	FS	Link5	8	0
6	3	17.2225	14	FS	Link6	10	4
7	6	145.7938	7	FS	Link7	5	2
8	8	154.3115	6	FS	Link8	6	0
9	5	389.7082	2	FS	Link9	9	7
10	4	40.0639	13	FS	NA	14	1
11	7	46.4706	12	FS	Link10	12	0
12	51	391.3391	1	FST	Link11	1	0
13	5	52.8906	11	FS	Link12	11	0
14	4	16.9345	15	FS	NA	15	0

embeddings that does not perform well for short and noisy text data set. Embedding methods did not work properly in short texts because tokens in a short text have a thin contextual relation between each other and this relation get worse due to misspelled and incomprehensible tokens. A sample result of doc2Vec applying hypermeter values *vector_size* = 300, *min_count* = 2 and *epochs* = 45 respectively is shown in Table IV that exhibits most similar tweet and second most similar tweet of a particular tweet that does not have any noticeable similarity with any of those tweet document whereas a document must show similarity with at least to itself. Similarity score of each tweet to the particular tweet is represented inside the parentheses in the first column. Again, we thought in a different way to apply LDA [23] to find some topics that may represent events. Since the tweet text is very short, almost all of the time a tweet does not represent more that one event. That is why, we decided to apply LDA [23] on the aggregated tweet texts from corresponding time intervals. This, approach also fails to show the expected result because of the incoherent nature of tweet text. Due to the space limitation of the we could not present in this work. Thus, we decided to apply very popular tfIdf vectorization because we found that words in a tweet text has a few semantic relation between each other and word co-occurrence is better option to apply in this domain. Consequently, we found a better performance by comparing the result with the previously applied approaches.

TABLE IV
SAMPLE RESULT OF DOC2VEC

Terms	***Texts***
Document	guides on fixing sql injections vulnerabilities sql injection technique exploits security vulnerability occurring database layer application the vulnerability present user input either
Most Similar (0.8027611970901489)	free vps server ddos protected hosting
Second Most Similar (0.6457577347755432)	cvnway just ddos server
Median (0.21316465735435486)	minibb bbfuncsearchphp table sql injection
Least (-0.4030833840370178)	ransomware weapon used cyber attacks elixir ng news source trust

2) Validation of the Approach: Table V shows the performance result of our proposed approach according to the evaluation methodology described in Section IV-F. The annotators annotate 301 and found total 20 events and 6 tweet clusters that are not events. On the other hand, the our algorithm found total 16 events. Now, 15 events out of 16 events are real events (True Positive) included in 20 ground truth but one event is False positive. So, the True Positive, False Positive, False Negative and True Negative rates are 75%, 16.67%, 25% and 83.33% respectively and we got a good precision value that is 93.75%. An interesting news is that we can only stream a very small amount of tweets per millisecond approximately 1% of the total tweet posted that is addressed in this web article [25]. So, the Twitter data itself only is not sufficient to detect all of the ongoing cyber security event and that is why we limit ourself to calculate recall score by keeping track published cyber security events in online. In Table III, we present the fifteen true positive events, along with their tweet count and their corresponding scores. We order the events by their corresponding scores and match with the annotators' annotations. The 4^{th} and the 7^{th} column of the Table III present proposed approach event ranking and annotators' ranking respectively and comparing the 4^{th} and the 7^{th} column of the table, we can see the annotators predictions are quite similar to our approach in case of event detection and event ranking. The validity of the detection approach can be checked by clicking the link mentioned in the 6^{th} column to see the reports published in authentic blogs and newspapers. The 5^{th} column represents the type of events detected by our algorithm. The sum squared error (SSE) of the event ranking of our approach and annotator's ranking is 86 by calculating the difference mentioned in the 8^{th} column of the table.

TABLE V
CONFUSION MATRIX OF THE ALGORITHM'S GENERATED RESULT

Total Population	***Ground Truth positive***	***Ground Truth negative***
Derived positive	True Positive=75%	False Positive=16.67%
Derived negative	False Negative=25%	True Negative=83.33%

VI. Conclusion

We presented a novel machine learning and text information extraction method for the detection of cyber threat events from tweets. We considered two types of such events, those that are novel, and those that are further developments of previously detected tweets. Furthermore, we proposed an approach for the ranking of cyber threat events based on an importance score computed based on the named entities and keywords in the text of tweets. We also impute influence to users in order to assign a weighted score to noun phrases in proportion to user influence and the corresponding event scores for named entities and keywords. To evaluate the performance of our proposals, we measure the efficiency and detection error rate for events over a specified time interval, relative to human annotator ground truth, and demonstrate the feasibility of its application in detecting cyber threat events from tweets. Future directions of this research include the extension of our current method for detection and ranking of sub-events in each cyber threat event. Moreover, the heuristics applied in this work are presented as proofs of concept, while leaving room for further enhancement and customization per user requirements. As further venue of future word, the methodology used for influence measurement of users can be extended via means such as meta-network modeling and link extraction of the dynamic social network of users that are active in the cybersecurity domain.

References

[1] Q. Li, A. Nourbakhsh, S. Shah and X. Liu, "Real-Time Novel Event Detection from Social Media," *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, San Diego, CA, 2017, pp. 1129-1139. doi: 10.1109/ICDE.2017.157

[2] Adam Marcus, Michael S. Bernstein, Osama Badar, David R. Karger, Samuel Madden, and Robert C. Miller. 2011.,"Twitinfo: aggregating and visualizing microblogs for event exploration.", *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11). ACM*, New York, NY, USA, 227-236. DOI: https://doi.org/10.1145/1978942.1978975.

[3] Sina Dabiri, Kevin Heaslip,"Developing a Twitter-based traffic event detection model using deep learning architectures", *Expert Systems with Applications*, Volume 118, 2019, Pages 425-439, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2018.10.017.

[4] Mittal, Sudip, Prajit Kumar Das, Varish Mulwad, Anupam Joshi, and Tim Finin. "Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities.", *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 860-867. IEEE Press, 2016.

[5] A. Edouard,"Event detection and analysis on short text messages", *Universit Cte d'Azur, 2017.*

[6] Rupinder Paul Khandpur, Taoran Ji, Steve Jan, Gang Wang, Chang-Tien Lu, and Naren Ramakrishnan. 2017."Crowdsourcing Cybersecurity: Cyber Attack Detection using Social Media", *In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17).* ACM, New York, NY, USA, 1049-1057. DOI: https://doi.org/10.1145/3132847.3132866

[7] K.-C. Lee, C.-H. Hsieh, L.-J. Wei, C.-H. Mao, J.-H. Dai, and Y.-T. Kuang,"Sec-buzzer: cyber security emerging topic mining with open threat intelligence retrieval and timeline event annotation", *Soft Computing*, vol. 21, no. 11, pp. 28832896, 2017.

[8] Sapienza, Anna, Sindhu Kiranmai Ernala, Alessandro Bessi, Kristina Lerman, and Emilio Ferrara. "Discover: Mining online chatter for emerging cyber threats." *Companion of the The Web Conference 2018 on The Web Conference 2018* , pp. 983-990. International World Wide Web Conferences Steering Committee, 2018.

[9] Quentin Le Sceller, ElMouatez Billah Karbab, Mourad Debbabi, and Farkhund Iqbal. 2017."SONAR: Automatic Detection of Cyber Security Events over the Twitter Stream." *Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES '17). ACM*, New York, NY, USA, Article 23, 11 pages. DOI: https://doi.org/10.1145/3098954.3098992

[10] Ifrim, Georgiana, Bichen Shi, and Igor Brigadir."Event Detection in Twitter using Aggressive Filtering and Hierarchical Tweet Clustering." *In SNOW-DC@ WWW*, pp. 33-40. 2014.

[11] Alan Ritter, Evan Wright, William Casey, and Tom Mitchell. 2015."Weakly Supervised Extraction of Computer Security Events from Twitter." *n Proceedings of the 24th International Conference on World Wide Web (WWW '15). International World Wide Web Conferences Steering Committee*, Republic and Canton of Geneva, Switzerland, 896-905. DOI: https://doi.org/10.1145/2736277.2741083

[12] https://github.com/behzadanksu/cybertweets

[13] TextRazor-2019;https://www.textrazor.com/

[14] Mihalcea, Rada, and Paul Tarau. "Textrank: Bringing order into text." *Proceedings of the 2004 conference on empirical methods in natural language processing*. 2004.

[15] Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. "The PageRank citation ranking: Bringing order to the web". *Stanford InfoLab*, 1999.

[16] Ester, Martin, Hans-Peter Kriegel, Jrg Sander, and Xiaowei Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise." *Kdd, vol. 96*, no. 34, pp. 226-231. 1996.

[17] H. Wu and R. Luk and K. Wong and K. Kwok. "Interpreting TF-IDF term weights as making relevance decisions. *ACM Transactions on Information Systems*, 26 (3). 2008.

[18] Wolf Garbe ¡wolf.garbe@faroo.com¿,"SymSpell 6.4", https://github.com/wolfgarbe/symspell

[19] Behzadan, Vahid, Carlos Aguirre, Avishek Bose, and William Hsu. "Corpus and Deep Learning Classifier for Collection of Cyber Threat Indicators in Twitter Stream". *2018 IEEE International Conference on Big Data (Big Data)*, pp. 5002-5007. IEEE, 2018.

[20] Radim rehurek and Petr Sojka"Software Framework for Topic Modelling with Large Corpora",*Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*,pages 45–50, May 22, 2010; DOI: http://is.muni.cz/publication/884893/en

[21] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E., "Scikit-learn: Machine Learning in Python" *Journal of Machine Learning Research*, volume 12, pp 2825–2830, 2011

[22] Le, Quoc, and Tomas Mikolov. "Distributed representations of sentences and documents." In *International conference on machine learning*, pp. 1188-1196. 2014.

[23] Blei, David M.; Ng, Andrew Y.; Jordan, Michael I (January 2003). Lafferty, John (ed.). "Latent Dirichlet Allocation". *Journal of Machine Learning Research.* 3 (45): pp. 9931022.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean."Efficient estimation of word representations in vector space". *CoRR*, abs/1301.3781, 2013.

[25] Andy Piper, "Potential adjustments to Streaming API sample volumes", *https://twittercommunity.com/t/potential-adjustments-to-streaming-api-sample-volumes/31628*, Feb 2, 2015.