Quantifying the Significance of Cybersecurity Text through Semantic Similarity and Named Entity Recognition

Otgonpurev Mendsaikhan[1], Hirokazu Hasegawa[2], Yamaguchi Yukiko[3] and Hajime Shimada[3]

[1]*Graduate School of Informatics, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Japan*

[2]*Information Strategy Office, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Japan*

[3]*Information Technology Center, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Japan*

Keywords: Cyber Threat, Semantic Similarity, NER, Text Analysis.

Abstract: In order to proactively mitigate the risks of cybersecurity, security analysts have to continuously monitor threat information sources. However, the sheer amount of textual information that needs to be processed is overwhelming and requires a great deal of mundane labor. We propose a novel approach to automate this process by analyzing the text document using semantic similarity and Named Entity Recognition (NER) methods. The semantic representation of the given text has been compared with pre-defined "significant" text and, by using a NER model, the assets relevant to the organization are identified. The analysis results then act as features of the linear classifier to generate the significance score. The experimental result shows that the overall system could determine the significance of the text with 78% accuracy.

1 INTRODUCTION

The digital age has enabled various opportunities in society and for business in general. However, these opportunities also impose different kinds of risks such as cyber attacks, data breach, loss of intellectual property, financial fraud, etc. One approach to mitigate those risks is the sharing of threat information via platforms such as the closed and open information-sharing communities as well as the threat feed generating vendors. The idea of sharing threat information stems from the assumption that an adversary that attacks a certain target is also likely to attack similar targets in the near future. While information sharing platforms have grown in popularity, the amount of shared threat information has grown tremendously, overwhelming human analysts and undermining the efforts to share threat information. In order to identify the significance of the shared information and relevance to their organizations, the analysts have to process considerable amounts of information and separate the actionable threat information from the noise.

Even though there are approaches that automatically share information between machines through structured information sharing such as Structured Threat Information Expression (STIX)[1] and its corresponding protocol Trusted Automated Exchange of Intelligence Information (TAXII), the need to process unstructured text reports that might be shared via email or forums still exists. For example, dark web forums provide valuable threat information, if the noise can be segregated, with less effort.

In our previous work we proposed a system to identify the threat information from publicly available information sources (Mendsaikhan et al., 2018). As a follow-up, in this paper we propose a novel approach to quantify the significance and relevance of the threat information in unstructured text by comparing the semantic representation of the text with known important text and identifying the IT assets through the Named Entity Recognition method. We considered the semantic similarity of the text and the number of entities as features of the threat information and fed them through a linear classifier to generate a confidence score that quantifies the significance of the text.

The main objective of this research is to seek a way to quantify the significance of a text document that can be customized to meet organizational needs using existing Natural Language Processing techniques and tools.

The specific contributions of the paper are as follows:

1. To propose a novel approach to analyze the text

[1]https://oasis-open.github.io/cti-documentation/

Mendsaikhan, O., Hasegawa, H., Yukiko, Y. and Shimada, H.
Quantifying the Significance of Cybersecurity Text through Semantic Similarity and Named Entity Recognition.
DOI: 10.5220/0008913003250332
In *Proceedings of the 6th International Conference on Information Systems Security and Privacy (ICISSP 2020)*, pages 325-332
ISBN: 978-989-758-399-5; ISSN: 2184-4356

documents to identify its significance

2. To prove the viability of the method through experiments
3. Custom train the Named Entity Recognition (NER) model to recognize IT related products

The remainder of this paper is organized as follows. Section 2 will review the related research and how this paper differs in its approach. In Section 3 we will briefly review our previous work on an autonomous system to generate cyber-threat related information. In Section 4 the implementation and evaluation of the Analyzer module of the proposed system and the corresponding experiment will be discussed. Finally, we will conclude by discussing future work to extend this research in Section 5.

2 RELATED WORK

There have been a number of attempts to automatically identify or extract cyber-threat related information from the dark web or any other publicly available information sources. Mulwad et al. described a prototype system to extract information about security vulnerabilities from web text using an SVM classifier before extracting the entities and concepts of interest from it (Mulwad et al., 2011). They tried to spot cybersecurity risks using knowledge from Wikitology, an ontology based on Wikipedia. Our approach differs in that we use common algorithms of Named Entity Recognition trained on a bigger dataset. Joshi et al. proposed a cybersecurity entity and concept spotter that uses the Stanford Named Entity Recognizer (NER), a Conditional Random Field (CRF) algorithm-based NER framework (Joshi et al., 2013). They focused on developing a more comprehensive data structure, whereas our approach is to identify a single label that is associated with a known IT asset.

More et al. proposed a knowledge-based approach to intrusion detection modeling in which the intrusion detection system automatically fetches threat-related information from web-based text information and proactively monitors the network to establish situational awareness. Their approach focused mainly on developing a cybersecurity ontology that could be understood by intrusion-detecting machines (More et al., 2012). Our research focused on building an autonomous system that assists the human operators by raising situational awareness.

Bridges et al. did the automatic labeling for an entity extraction from a cybersecurity corpus consisting of 850,000 tokens (Bridges et al., 2013). Their dataset was the inspiration for preparing a similar dataset from the whole archive of CVE descriptions. Jones et al. attempted to extract cybersecurity concepts using Brin's Dual Iterative Pattern Relation Expansion (DIPRE) algorithm, which uses a cyclic process to iteratively build known relation instances and heuristics for finding those instances (Jones et al., 2015). Our approach differs by deploying an off-the-shelf software solution instead of creating a new method.

Dionísio et al. developed a system to detect cyber-threats from Twitter using deep neural networks (Dionísio et al., 2019). Their work has many similarities with our work, e.g. collecting relevant threats from Twitter feeds and identifying the assets through Named Entity Recognition. Our approach differs by extracting cybersecurity-related documents and then prioritizing the relevance of each document.

3 BACKGROUND

Since this paper continues from our previous research, the background of the research and proposed system architecture will be briefly introduced in the subsequent sections.

3.1 Proposed System Overview

In our previous work we proposed a system to identify threat information from publicly available information sources (Mendsaikhan et al., 2018). The proposed system architecture is depicted in Figure 1.

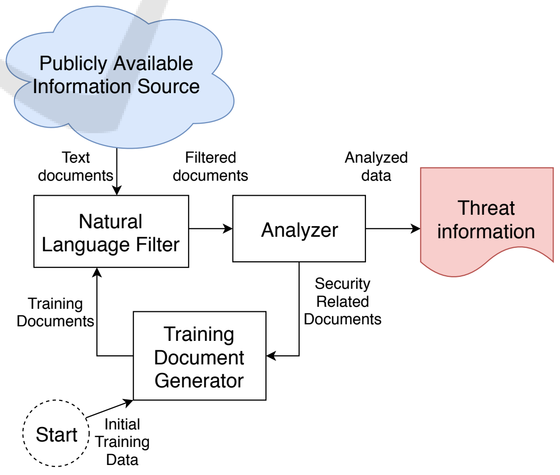

Figure 1: Overview of proposed system.

The proposed system would scan the publicly available information sources on the Internet to create situational awareness and to assist the security analyst in identifying risks and threats posed to his organization. This method utilizes the Natural Language Filter module to identify and filter the security-related text documents. The organization specific textual data is collected as initial training data and fed to the Training Document Generator module to prepare the training documents. The Natural Language Filter module is trained by these documents and filters the security-related documents. The collected and filtered documents are analyzed using the Analyzer module to generate meaningful threat information for the human operators. The documents that have been marked as true positive by the Analyzer module are fed back to the Training Document Generator to improve the performance of the Natural Language Filter.

3.2 Previous Work

In our previous work we implemented the Natural Language Filter module of the proposed system (Mendsaikhan et al., 2019). We proposed to utilize a neural embedding method called Doc2Vec (Le and Mikolov, 2014) as a natural language filter for the proposed system. With the cybersecurity-specific training data and custom preprocessing, we were able to train a Doc2Vec model and evaluate its performance. According to our evaluation, the Natural Language Filter was able to identify cybersecurity-specific natural language text with 83% accuracy.

As a continuation of our previous work, this paper focuses on the implementation of the Analyzer module as described in the subsequent sections.

3.3 Analyzer Module

We believe the similarity score of a text document with different types of documents at a semantic level, along with the relevant named entities mentioned, could determine the potential significance of the threat information in the text format. The theory is, if a given text document is semantically similar to a certain class of documents and also mentions specific IT assets in the form of named entities, the text might be relevant and significant to the organization. In order to prove whether our approach is valid, we designed a system that consists of the following components.

1. Semantic Analyzer
2. Named Entity Analyzer
3. Significance Score Calculator

The high level overview of the Analyzer module is depicted in Figure 2.

The collected threat information is analyzed concurrently by a Semantic Analyzer and Named Entity

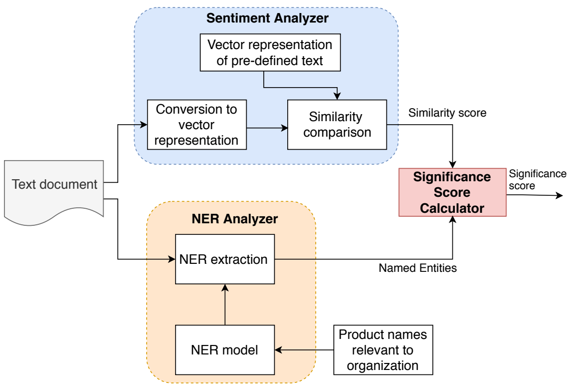
Sentiment Analyzer
Vector representation of pre-defined text
Conversion to vector representation
Similarity comparison
Similarity score
Text document
Significance Score Calculator
Significance score
NER Analyzer
NER extraction
Named Entities
NER model
Product names relevant to organization

Figure 2: Overview of Analyzer module.

Analyzer. The Semantic Analyzer converts the document into vector representations and compares it with different types of document. The highest similarity score for each type of document is fed into a Significance Score Calculator. Meanwhile, the Named Entity Analyzer extracts the entities that are relevant to the organization and pass them into the Significance Score Calculator. Finally, the Significance Score Calculator computes the score that the human analyst can judge to decide whether to consider it for further analysis.

Each component of the Analyzer module is discussed in detail in the subsequent sections.

3.3.1 Semantic Analyzer

The semantic analysis of the text document refers to extracting the lexical meaning of a text independent of its written language. Since computers can work only with numbers, computational linguistics achieve semantic analysis by representing text in vector space and assigns different meanings of the text in different dimensions of the vector. For example, the word "bank" could mean a financial institution as well as geographical terrain adjacent to a river (as in river bank). When the word "bank" is represented in vector space, each meaning would be represented by different components of a same vector depending upon the context. Once the text is represented in vector space, one way of performing the semantic analysis on the text document is to compare the vector representation of it with another vector which represents pre-defined "significant" text. Comparing the vector representations of different texts is called semantic similarity and the distance between the vectors would represent the closeness of the semantic meaning between them.

We believe semantic similarity could be used to define the significance of the text by comparing vector representations of the given text with a pre-defined "significant" text.

3.3.2 Named Entity Analyzer

Named Entity Recognition (NER) is part of the information extraction task in Natural Language Processing. NER models are trained to identify real world entities such as people, locations, organizations, etc. Commonly known approaches to implement the NER model consist of rule or pattern based approaches such as identifying the patterns of entities with regular expression or statistics or machine learning based algorithms such as Conditional Random Fields (CRF).

Since every organization has different priorities and are exposed to different cyber risks, we believe the Named Entity Analyzer would help to personalize the significance of the textual document. In other words, by specifying a list of their IT assets in the Named Entity Analyzer, the organization would be able to customize the significance of incoming text; thus, they receive the threat information that fits their requirements. Conceptually, the organization specifies the IT products that are relevant to them and the NER model is trained to find similar IT assets in the given text, as shown in Figure 2.

3.3.3 Significance Score Calculator

The Significance Score Calculator (SSC) is a function that outputs a fixed range of numbers based on the given inputs. The inputs consist of the following items.

- Highest similarity score with the pre-defined significant text
- Number of named entities found in the document that are of interest to the organization

These inputs would serve as features to be extracted from the threat information document to classify whether the document is significant or not. Ideally, SSC would be a linear classification system that produces quantitative numbers which represent the confidence of specific item belonging to a significant class.

4 IMPLEMENTATION AND EVALUATION

To verify the viability of the proposed system, the experiment has been conducted by implementing the proposed components using common open source libraries.

4.1 Implementation of Semantic Analyzer

In order to perform semantic analysis through textual similarity, the given text is converted into numerical vectors, also known as embeddings. Conventionally, vector embeddings were achieved through shallow algorithms such as Bag of Words (BoW) or Term Frequency-Inverse Document Frequency (TF-IDF). These approaches have been preceded by predictive representation models such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) etc. Since the utilization of deep neural networks has been proven to be superior in different fields, various studies have adopted deep neural models to embed the text into vector space, such as Facebook's InferSent[2] and Universal Sentence Encoder (USE) from Google Research. Perone et al. evaluated different sentence embeddings and Universal Sentence Encoder outperformed InferSent in terms of semantic relatedness and textual similarity tasks (Perone et al., 2018). Therefore, for the purpose of this research, Universal Sentence Encoder has been implemented as the Semantic Analyzer.

4.1.1 Dataset

Phandi et al. proposed a shared task to classify relevant sentences, predict token labels, relation labels and attribute labels of malware-related text at the International Workshop on Semantic Evaluation 2018 (Phandi et al., 2018). In the task proposal they have compiled the largest publicly available dataset of annotated malware reports, which is called MalwareTextDB and consists of 85 Advanced Persistent Threat (APT) reports that contain 12,918 annotated sentences. The focus of their work was on marking the words and phrases in malware reports that describe the behavior and capabilities of the malware. For this purpose, the authors utilized Mitre's Malware Attribute Enumeration and Characterization (MAEC) vocabulary.

MAEC is a structured language for encoding and sharing high-fidelity information about malware based upon various attributes[3]. Kirillov et al. proposed MAEC in their paper (Kirillov et al., 2010) as an effort to characterize malware based on their behaviors, artifacts, and attack patterns. MAEC authors developed a vocabulary that enumerates and describes the common terminologies used in malware reports.

Using MAEC vocabulary, Phandi et al. constructed four different classes that could describe

[2]https://github.com/facebookresearch/InferSent
[3]https://maecproject.github.io/about-maec/

the actions and capabilities of the malware, specifically *ActionName*, *Capability*, *StrategicObjectives*, and *TacticalObjectives*. Each token from the sentences of the APT reports have been annotated and labeled as either *Action*, *Subject*, *Object* or *Modifier*. Each *Action* token that expresses the malware action or capability has been assigned to one or more classes of *ActionName*, *Capability*, *StrategicObjectives* and *TacticalObjectives*.

Each class represents the different actions and capabilities of the malware. For example, if the words "prevent sandbox analysis" are present in the sentence in the context of the malware's anti-sandbox technique, then the sentence is annotated with the label AntiBehavioralAnalysis and considered in the StrategicObjectives class, whereas if the word "exfiltrate" is present, then the sentence is annotated with the DataExfiltration label and assigned to the TacticalObjectives class.

We believe the actions and capabilities of the malware are of utmost importance to the human analysts; therefore, if the given text is semantically similar to pre-defined text describing malware actions and capabilities, the significance of that text should be considered high. MalwareTextDB2.0 has been utilized as pre-defined "significant" text and extracted four different classes of text as described above. For each APT report file contained in MalwareTextDB2.0, starting from the beginning of a sentence annotated as malware action or capability, 512 characters have been extracted to preserve the context. Even though it is claimed that USE can work on varying lengths of text, we observed that better semantic similarity is obtained when texts of the same length are compared. Hence, in order to fix the length of the text, we have arbitrarily chosen 512 characters. If the same sentence is annotated with multiple classes, the least frequent class assignment are considered to evenly distribute. After removing the duplicates, in total 1,927 sentences have been extracted into four different classes, as shown in Table 1.

Table 1: Class distribution.

Class	Sentences extracted
ActionName (AN)	275
StrategicObjectives (SO)	394
TacticalObjectives (TO)	231
Capability (Cap)	1027
Total	1927

From each class, 100 mutually exclusive sentences have been randomly selected to act as pre-defined Reference text. The sentences of each class have been compared with sentences from other classes in the Reference text and maximum cosine similarity between the classes are generated, as shown in Table 2.

Table 2: Similarity comparison between classes.

	AN	SO	TO	Cap
AN	1.0	0.9626	0.9672	0.9747
SO		1.0	0.9901	0.9275
TO			1.0	0.8757
Cap				1.0

From Table 2, it can be seen that the classes of Reference text are semantically very similar to each other. This is because all the extracted sentences came from similar types of documents, i.e. malware reports, contained in MalwareTextDB2.0.

The remaining 1,527 sentences have been kept in order to train and evaluate the overall system, as will be discussed in Section 4.3.1.

4.1.2 Universal Sentence Encoder

In the paper by Cer et al. transformer-based and deep averaging network (DAN)-based models for encoding sentences into embedding vectors have been introduced (Cer et al., 2018). The USE models take variable length English sentences as input and produce 512 fixed dimensional vector representations of the sentences as output. Both the models have been pre-trained using Wikipedia, web news, web question-answer pages and discussion forums[4].

Since the sentence embeddings from USE produce good task performance with little task-specific training data, a DAN-based sentence encoder has been utilized for this research in order to find textual similarity between the texts in vector space, thus performing a semantic analysis. The DAN-based sentence encoder model makes use of a deep averaging network whereby input embeddings for words and bi-grams are first averaged together and then passed through a feedforward deep neural network to produce sentence embeddings with minimal computing resource requirements.

Prior to the semantic analysis process, vector representations of every entry in the Reference text are pre-generated using USE and stored in separate repositories. Similar to the Reference text, 512 characters from the start of the input text have been extracted and converted into a vector representation using USE. Consequently, cosine similarity is computed with each entry in the Reference text and the highest similarity score for each class is considered as input into the Significance Score Calculator.

[4]https://tfhub.dev/google/universal-sentence-encoder/2

4.2 Implementation of Named Entity Analyzer

We believe by identifying the relevant named entities mentioned in the text document, it is possible to prioritize a variety of threat information as per the organizational need. To implement the Named Entity Analyzer, a NER model is required that could be easily retrained with low computing resources. Therefore, off-the-shelf open source library spaCy has been considered for this research. The spaCy NER model has 85% accuracy, which is slightly lower than the state-of-the-art model but more efficient on generic CPU[5]. The spaCy NER model has been trained to identify the organization specific IT assets under the label "IT-Product".

4.2.1 Dataset

Training a domain-specific NER model is difficult due to the lack of annotated training data in the specific domain. Fortunately, the National Vulnerability Database (NVD)[6] of the National Institute of Standards and Technology (NIST) provides the Common Vulnerabilities and Exposures (CVE) in a structured format that can be used to train the NER model. Also, the Common Product Enumeration (CPE) of the NVD provides structured naming conventions for commonly used software and hardware products. The CPE dictionary contains vendor name, product name, product version, environment etc., in a Uniform Resource Identifier (URI) format, which can be extracted from the CVE description to be used as training data for the NER model.

By utilizing the CVE descriptions and CPE dictionary from the NVD, a total of 109,635 CVE descriptions, as of 8th July 2019, have been retrieved. For the purpose of this paper, we did not specify any specific product, and all the products and companies in the CPE dictionary have been used to train the model. Ninety percent of the total collected data has been used to train the spaCy NER model and the remaining 10% of documents has been used to test the trained model.

4.2.2 NER Model

spaCy's Named Entity Recognition system features a sophisticated word-embedding strategy using subword features and "Bloom" embeddings, a deep convolutional neural network with residual connections, and a novel transition-based approach to named-entity parsing. spaCy's en-core-web-lg model, which have been trained on OntoNotes in English and also contains GloVe vectors trained on Common Crawl[7], has been utilized as a pre-trained model. Named Entity Analysis is achieved through further training of the en-core-web-lg model with the training data obtained from the NVD and adding a label of "ITProduct" to identify the IT assets.

After custom training the model, its performance is evaluated with a test dataset of 10,962 documents that contains a total of 19,452 entities with the label "ITProduct". The trained model identified 13,713 entities correctly (True Positive) and missed 5,739 entities (False Negative) and misidentified 3,446 entities (False Positive). Using those numbers the performance was compared with default en-core-web-lg model, as shown in Table 2.

Table 3: Performance Comparison of NER models.

Model	Precision	Recall	F1 Score
Default model	87.03	86.20	86.62
Custom model	79.91	70.49	74.91

The custom-trained model shows poor performance compared to the default model. By fine-tuning the parameters, it would be possible to achieve better result in future studies.

4.3 Implementation of Significance Score Calculator

The evaluation of the overall system could be determined by the classification results of the predetermined significant and non-significant texts. Each output of the Semantic Analyzer and Named Entity Analyzer is inputted into the Significance Score Calculator (SSC) to generate classification result that can be used to evaluate the overall performance of the system. The Support Vector Machine is chosen as linear classifier for the SSC due to its efficiency with low amounts of training data and the confidence score output.

4.3.1 Dataset

Since the main application of the SVM algorithm is binary classification, the best result is obtained when a balanced dataset of positive and negative examples is used. Hence, the remaining 1,527 extracted sentences of MalwareTextDB2.0 are considered as positive, i.e. "significant", examples. For the negative, i.e. "insignificant" examples, the same number of documents from the StackExchange discussion forum has

[5]https://spacy.io/usage/facts-figures

[6]http://nvd.nist.org

[7]https://spacy.io/models/en#en_core_web_lg

been utilized. StackExchange[8] is a network of question and answer websites on various topics. As part of our previous work, a total of 841,311 security-related text documents were collected and from them 1,527 randomly selected documents have been considered as negative examples. Both the positive and negative examples are from security-related text, though the positive examples consist of sentences contained in official malware reports, whereas the negative examples consist of casual conversations around any security topic on informal discussion forums. The overall dataset construction process is shown in Figure 3.

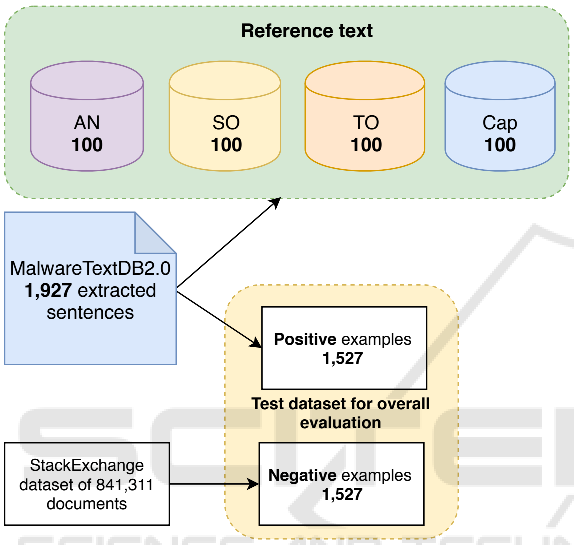
Reference text
AN 100
SO 100
TO 100
Cap 100
MalwareTextDB2.0 1,927 extracted sentences
Positive examples 1,527
Test dataset for overall evaluation
StackExchange dataset of 841,311 documents
Negative examples 1,527

Figure 3: Dataset construction process.

Both the positive and negative examples have been passed through the Semantic Analyzer and Named Entity Analyzer, respectively, to extract their features. Each entry in the dataset has the following five features extracted:

- Number of named entities labeled as "ITProduct" that have been found in the document
- Highest similarity score with the ActionName class from the Reference text
- Highest similarity score with the StrategicObjectives class from the Reference text
- Highest similarity score with the TacticalObjectives class from the Reference text
- Highest similarity score with the Capability class from the Reference text

Once all the features have been extracted per entry, the positive and negative examples are randomly mixed and fed to the SVM-based Significance Score Calculator.

[8]https://stackexchange.com/

4.3.2 Support Vector Machine

The Support Vector Machine (SVM) is a popular supervised machine learning algorithm mostly used for classification and regression problems. SVM classifiers separate the datasets by finding a line or hyperplane by computing the closest points to both classes of data. These points are called Support Vectors and the distance between the line to dataset is called the margin. The SVM algorithm maximizes the margin, thus giving the optimal classification between datasets.

The features mentioned in Section 4.3.1 have been fed to sklearn[9]'s implementation of the SVM classifier. The sklearn's SVM classifier can have different kernels in form of functions depending upon the shape of the hyperplane. Preliminary experiments revealed that the Radial Basis Function (RBF) has the best performance in our dataset; therefore, the RBF mode of the SVM kernel was chosen and every other hyperparameter were kept as default. sklearn's SVM classifier can have three modes of output such as

- Binary classification into two classes
- Probability of item belonging to either classes
- Confidence score of the item to belong either class

Since the objective of the research is to generate quantitative numbers that represent the significance of the text, this implementation suits our needs. However, in order to evaluate the viability of the proposed method, an experiment with the test dataset was conducted to generate the performance indicators, such as Accuracy and F1 score.

4.4 Evaluation Results

As mentioned in Section 4.3.1, a balanced dataset of 3,054 documents was used to train and test the SVM model using a 10-fold cross validation method. The maximum, minimum and average performances of the 10-fold cross validation in terms of Accuracy are shown in Table 4.

Table 4: Evaluation result.

	Prec.	Rec.	F1 Score	Accuracy
Max	81.54	84.56	83.03	81.69
Min	70.70	78.72	74.49	75.16
Average	76.51	81.53	78.92	78.28

The averages of the evaluation metrics show that the confidence score generated by the SVM classifier could be used as the significance score for this setup with 78% accuracy.

[9]https://scikit-learn.org/stable/modules/svm.html

To justify the choice of SVM classifier for the SSC, the same data is used to train and classify the Decision tree-based classifier. The average results of the 10-fold cross validation for SVM-based and Decision tree-based classification experiments are shown in Table 5.

Table 5: Performance comparison of SVM and Decision tree based classifier.

	Prec.	Rec.	F1 Score	Accuracy
SVM	76.51	81.53	78.92	78.28
Dec. tree	73.69	74.40	74.03	73.96

The comparison result shows that the SVM classifier performs better than the Decision tree-based classifier, which confirms the choice of SSC.

Overall, the experiment result seems to be compelling evidence that the significance score of the cyber-threat information could be calculated by an SVM classifier using the features generated by semantic textual similarity and a custom-trained NER model.

5 CONCLUSION

In this paper we proposed a novel approach to quantify the relevance and significance of cyber-threat information in text format by extracting the features such as maximum similarity scores with a pre-defined "significant" text and a number of relevant named entities. The experiment result shows the potential of our approach with 78% accuracy.

As sentences used in classes of Reference text have come from the same source and are homogeneous in nature, the semantic similarities of the different classes of Reference text are too close, as shown in Table 2. This drawback affects the features of the input text, thus reducing the overall performance. By selectively using various sources and assigning different weight scores depending upon the relevance to the classes of Reference text, this problem could be solved. Also, the Named Entity Analyzer contributes only one feature to the overall model prediction; therefore, changing the design of the experiment to generate more features from named entities could improve the operation of the Named Entity Analyzer.

Regarding future work, we would improve our experiment's design by accommodating the changes mentioned and also using a new NER model by utilizing different algorithms and extending the scope of the entities further than the single label of "ITProduct".

REFERENCES

Bridges, R. A., Jones, C. L., Iannacone, M. D., and Goodall, J. R. (2013). Automatic labeling for entity extraction in cyber security. *CoRR*, abs/1308.4941.

Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder. *CoRR*, abs/1803.11175.

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

Jones, C. L., Bridges, R. A., Huffer, K. M. T., and Goodall, J. R. (2015). Towards a relation extraction framework for cyber-security concepts. *CoRR*, abs/1504.04317.

Joshi, A., Lal, R., Finin, T., and Joshi, A. (2013). Extracting cybersecurity related linked data from text. In *Proceedings of 7th International Conference on Semantic Computing*.

Kirillov, I., Chase, P., Beck, D., and Martin, R. (2010). Malware attribute enumeration and characterization. White paper, The MITRE Corporation, Tech.

Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents. *CoRR*, abs/1405.4053.

Mendsaikhan, O., Hasegawa, H., Yamaguchi, Y., and Shimada, H. (2018). Mining for operation specific actionable cyber threat intelligence in publicly available information source. In *Proceedings of Symposium on Cryptography and Information Security*.

Mendsaikhan, O., Hasegawa, H., Yamaguchi, Y., and Shimada, H. (2019). Identification of cybersecurity specific content using the doc2vec language model. In *Proceedings of IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations*.

More, S., Matthews, M., Joshi, A., and Finin, T. (2012). A knowledge-based approach to intrusion detection modeling. In *Proceedings of 2012 IEEE Symposium on Security and Privacy Workshops*.

Mulwad, V., Li, W., Joshi, A., Finin, T., and Viswanathan, K. (2011). Extracting information about security vulnerabilities from web text. In *Proceedings of International Conferences on Web Intelligence and Intelligent Agent Technology*.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Perone, C. S., Silveira, R., and Paula, T. S. (2018). Evaluation of sentence embeddings in downstream and linguistic probing tasks. *CoRR*, abs/1806.06259.

Phandi, P., Silva, A., and Lu, W. (2018). Semeval-2018 task 8: Semantic extraction from cybersecurity reports using natural language processing (SecureNLP). In *Proceedings of The 12th International Workshop on Semantic Evaluation*.