Electronic Preprint for Journal of Information Processing Vol.27

Regular Paper

An Attempt to Read Network Traffic with Doc2vec

Mamoru Mimura[1,a)]

Received: March 18, 2019, Accepted: August 9, 2019

Abstract: Detecting new malicious traffic is a challenging task. There are many behavior-based detection methods which extract the features of malicious traffic. However, many previous methods require knowledge of how to extract feature vectors. If attackers modify the attack techniques, these previous methods may have to extract new feature representation to detect them. To address this problem, neural networks can be applied to perform feature learning. Doc2vec is one of these models that learn fixed-length feature representation from variable-length documents and has been applied to proxy logs. However, some attackers still use protocols other than http or https. In this paper, we extend the previous method to a generic detection method which supports any protocol. The key idea of this research is reading network packets as a natural language. In our method, a protocol analyzer reads network packets, and summarizes the traffic. Our method extracts the feature representation from the summary with Doc2vec. We apply several classifiers to the automatically extracted feature representation, and classify traffic into benign and malicious traffic. In the fundamental experiment, the best F-measure achieves 0.98 in the timeline analysis and 0.97 in the cross-dataset validation. Furthermore, we generate imbalanced datasets which simulate actual network traffic. In the practical experiment, the best F-measure achieves 0.82 in the timeline analysis and 0.73 in the cross-dataset validation.

Keywords: intrusion detection, neural network, Bag-of-Words, Word2vec, Doc2vec, support vector machine, random forests, Multi-Layer Perceptron

1. Introduction

In recent years, Drive-by Download attacks (DbD attack) and spear phishing attack are the main attack techniques on the Internet. In general, basic intrusion detection techniques on a network are roughly classified into pattern-matching-based methods and methods using blacklists. The pattern-matching-based methods are effective if the malicious traffic contains a unique string pattern. The IDS signatures are described with fixed strings or regular expressions. However, recent malware communicates via a standard protocol to imitate benign traffic. Some queries do not contain the specific strings. Therefore, it is difficult to describe the signatures. In this case, the signatures can be described with the malicious destination server (e.g., Landing site, C&C server) address. These addresses are also used as the blacklist on a firewall or a proxy server. However, attackers can easily change the destination servers to evade detection by network devices. Moreover, some attackers use compromised hosts as stepping stones. To decide whether a host is malicious or not, we have to investigate the context carefully. Therefore merely labeling each host will not prove effective. Hence, using the blacklist does not play a critical role. In addition, the malicious server address has to be already-known before the cyberattack. Thus, detecting new malicious traffic is a challenging task.

There are many behavior-based detection methods which extract the features of malicious traffic. These methods extract the features of DbD attacks [1] or C&C traffic [2], [3], [4], and attempt to detect new malicious traffic. Many previous methods, however, require knowledge of how to extract feature vectors. Moreover, we may have to add other features manually to learn new malicious traffic. In fact, new features are devised by researchers one after another. Hence, if attackers modify the attack techniques, these previous methods may have to extract new feature representation to detect them. Because these detection techniques tend to be optimized for each attack technique. Furthermore, this requires expert knowledge and skills. The problems with these previous methods are summarized as follows: require learning other features manually, require expert knowledge and skill. To address these problems, neural networks can be applied to perform feature learning. Doc2vec is one of these models that learn fixed-length feature representation from variable-length documents and has been applied to proxy logs [5]. However, some attackers still use protocols other than http or https. For instance, recent WannaCry ransomware uses Server Message Block (SMB) to compromise Windows machines, load malware, and propagate to other machines in a network. SMB is a transport protocol used by Windows computers for a wide variety of purposes such as file or printer sharing. To detect unauthorized access traffic with any protocol, a generic detection method which supports any protocol is required.

In this paper, we extend the previous method [5] to a generic detection method which supports any protocols. The key idea of this research is reading network packets as a natural language. To read network packets, our method uses TShark which is a command line-based network protocol analyzer. TShark captures network packets from a live network or reads packets from a previously saved capture file, and prints a decoded form of those packets to the standard output. Our method extracts words from the

[1] National Defense Academy, Yokosuka, Kanagawa 239–8686, Japan
[a)] mim@nda.ac.jp

output, and constructs a Doc2vec model. To evaluate our method, we use captured malicious traffic which was downloaded from the website MALWARE-TRAFFIC-ANALYSIS.NET [6]. In the fundamental experiment, the best F-measure achieves 0.98 in the timeline analysis and 0.97 in the cross-dataset validation. Furthermore, we generate imbalanced datasets which simulate actual network traffic. In the practical experiment, the best F-measure achieves 0.82 in the timeline analysis and 0.73 in the cross-dataset validation.

The main contributions of this paper are as follows:

(1) Extends the previous method [5] to a generic detection method which supports any protocols.
(2) Effectively uses Doc2vec in an application layer to classify traffic [7].
(3) Verify the performance of the proposed method on imbalanced datasets.

The rest of the paper is organized as follows. Next section briefly discusses related works and clarifies the difference the difference between our method and previous methods. Section 3 describes Natural Language Processing (NLP) techniques which include Paragraph Vector. Section 4 proposes a generic detection method based on the NLP techniques. Section 5 shows experimental results applying the proposed method to the multiple datasets. Section 6 discusses the results, and reveals the performance and the effectiveness of our method.

2. Related Work

In general, the main studies of network intrusion detection include signature-based detection and behavior-based detection. Signature-based detection relies on an existing signature database to detect known malicious traffic and seldom detects new malicious traffic. Therefore, many behavior-based detection methods are proposed. For example, some methods focused on the traffic classification from packet traces [8], [9], [10], [11], [12]. However, analyzing packets is proving difficult on broadband networks. The alternative approach is classification based on network logs such as DNS records, NetFlow or proxy server logs. There are several methods which use NetFlow [2], [3], DNS records [13], [14], [15], [16] and proxy server logs [4], [5], [17], [18], [19], [20], [21], [22], [23], [24]. Some approaches focus on NN to detect basic network attacks [25], [26], [27]. However, recent malware is sophisticated and communicates via a standard protocol to imitate normal communication. Furthermore, some attackers use compromised hosts as stepping stones. Therefore, detecting recent malicious traffic from logs is becoming a challenging task. Hence, many methods use additional contents to distinguish malicious traffic from seemingly benign traffic. For instance, some methods analyze JavaScript code to detect them [1].

These previous methods utilize the features of DbD attacks or C&C traffic well. Many previous methods are optimized for each attack technique, and the adaptability is limited. In addition, many previous methods using machine learning technique demand devising feature vectors to distinguish malicious traffic. In other words, the essence of the previous works was how to extract feature vectors from network traffic, logs, and contents. Our method is fundamentally different and based on the other viewpoint. Our method attempts to detect a variety of malicious traffic with a simple technique. Hence, our method does not require devising feature vectors. Because our method learns the difference between benign and malicious traffic automatically with NN.

3. Natural Language Processing (NLP) Technique

3.1 Bag-of-Words (BoW)

To calculate various measures to characterize a text, we have to convert the text into a vector. Bag-of-Words (BoW) is a simplifying representation used in NLP. In this model, a sentence is represented as the bag of its words, disregarding grammar and even word order but keeping multiplicity. BoW is commonly used in document classification methods where the frequency of each word is used as a feature for training a classifier. The most common type of features calculated from BoW is the number of times a term appears in the sentence. However, term frequencies are not necessarily the best representation for the sentence. BoW cannot represent grammar, word order, and word meaning. Furthermore, the feature dimension is defined by the vocabulary size of the data set. Therefore, we have to consider how to reduce the dimensionality for practical use.

3.2 Word2vec

Traditional NLP techniques cannot represent grammar, word order, and word meaning. Word2vec [28] is a model that produces word embedding. Word embedding is the collective name for a set of language modeling and feature learning techniques in NLP where words from the vocabulary are mapped to vectors of real numbers. This model is a shallow two-layer neural network that is trained to reconstruct the linguistic contexts of words. This model takes as its input a large corpus of text and produces a vector space, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to each other in the space. Word2vec is based on the distributional hypothesis, which signifies that the meaning of a word can be gauged by its context. Thus, if two words occur in the same position in two sentences, they are strongly related by semantics or syntax. Word2vec utilizes two algorithms to produce a distributed representation of words. One is Continuous-Bag-of-Words (CBoW), and the other is skip-gram. In the CBoW algorithm, the model predicts the current word from a window of surrounding context words. In the skip-gram algorithm, the model uses the current word to predict the surrounding window of context words. Word2vec allows to calculate the semantic similarity between two words and infer similar words semantically. Word2vec is a model that merely produces word embedding. To calculate the semantic similarity between two documents, this model has to be extended.

3.3 Paragraph Vector (Doc2vec)

An extension of Word2vec to construct embedding from entire documents has been proposed [29]. This extension is called Doc2vec or Paragraph2vec and has been implemented. Doc2vec

is based on the same distributional hypothesis, which signifies that the meaning of a sentence can be gauged by its context. Thus, if two sentences occur in the same position in two paragraphs, they are very much related either in semantics or syntactic in the same way. Doc2vec utilizes two algorithms to produce Paragraph Vector which is a distributed representation of entire documents. One is Distributed-Memory (DM), and the other is Distributed-Bag-of-Words (DBoW). DM is the extension of CBoW, and the only change in this model is adding a document ID as a window of surrounding context words. DBoW is the extension of skip-gram, and the current word was replaced by the current document ID. Doc2vec allows to calculate the semantic similarity between two documents and infer similar documents semantically. Some implementations also support inference of document embedding on new documents. This function is important to develop a practical system to classify new traffic. Because, new traffic might include many new words.

4. Proposed Method

4.1 Key Idea

The key idea of our method is reading network packets as a natural language. In order to apply NLP techniques, network packets have to be translated and separated into words. However, reading network packets is proving difficult on broadband networks. Therefore, we need a lightweight translation method. To translate network packets into a language, we use TShark [30] a network protocol analyzer, which captures and decodes packet data from a network. TShark displays a summary line for each received packet. The summary line consists of some fields such as source and destination IP address, protocol, size and basic contents. Our method uses these fields as separate words.

Figure 1 shows summary lines in a transport layer. In a transport layer, our method uses protocol, port number, size, and flags. Our method ignores other parameters, because there is too wide of a variety to process.

Figure 2 shows summary lines in an application layer. In an application layer, our method uses all fields after protocol. Furthermore, our method separates FQDN (Fully Qualified Domain Name) by “dot” (.). Then we can derive the top level domain name, sub domain name, and so on, which means the country, the organization, the use or the purpose (e.g., www, mail). Our method separates path by “slash” (/),“dot” (.), “question mark” (?), “equal” (=), and “and” (&). Then, we can derive the directory name, the file name, the extension from the path. We can also derive the variable names and the values from the query string, which are used in the running program on the server. Our method also separates IP addresses by “dot” (.), which are included in the fields after protocol. Thus, our method does not use any raw FQDNs or IP addresses. Therefore, our method does not depend on the specific environment. Note that these fields are not limited to http or https. Thus, our method is available in any protocol which TShark can analyze in an application layer.

4.2 Assumption

The purpose of our method is detecting new malicious traffic. The purpose is not filtering malicious summary lines. Nowadays,

```
192.168.204.175 → 206.188.192.114 TCP 66 49380 → 80 [SYN] Seq=0 Win=8192
Len=0 MSS=1460 WS=4 SACK_PERM=1
206.188.192.114 → 192.168.204.175 TCP 60 80 → 49380 [SYN, ACK] Seq=0 Ack=1
Win=64240 Len=0 MSS=1460
192.168.204.175 → 206.188.192.114 TCP 60 49380 → 80 [ACK] Seq=1 Ack=1
Win=64240 Len=0
```

Fig. 1 Summary lines in a transport layer.

```
10.180.0.14 → 10.180.0.254 DNS 82 Standard query 0xe854 A www.antiqueceramics.tk
10.180.0.14 → 10.180.0.254 DNS 82 Standard query 0xe854 A www.antiqueceramics.tk
10.180.0.254 → 10.180.0.14  DNS 371 Standard query response 0xe854 A
www.antiqueceramics.tk CNAME antiqueceramics.tk A 195.20.34.1 A 195.20.34.2 NS
d.ns.tk NS c.ns.tk NS b.ns.tk NS a.ns.tk A 194.0.41.1 AAAA 2001:678:5c::1 A 194.0.39.1
AAAA 2001:678:54::1 A 194.0.40.1 AAAA 2001:678:58::1 A 194.0.38.1 AAAA
2001:678:50::1
10.180.0.14 → 195.20.34.1  HTTP 316 GET / HTTP/1.0
195.20.34.1 → 10.180.0.14  HTTP 60 HTTP/1.0 203 Non-Authoritative Information
(text/html)
```

Fig. 2 Summary lines in an application layer.

attackers often use many compromised websites to imitate benign traffic. Hence, it is difficult to decide whether a summary line is benign or malicious independently. To decide whether a summary line is malicious or not, we have to investigate the context. We can investigate the context from sequential summary lines in network traffic. In this paper, the context is synonymous with sequential summary lines. To investigate the context, we define malicious traffic as a paragraph which contains malicious summary lines. The paragraph consists of summary lines which have the same number of rows. As a matter of course, a paragraph contains multiple summary lines. Hence, our method attempts to detect malicious traffic which might contain benign summary lines. We assume that these benign summary lines are related to the malicious summary lines. Our method does not extract only malicious summary lines. Our method investigates a paragraph comprehensively and determines whether a paragraph is malicious or not. Our method assume that the operators will perceive intrusions based on the result from classifiers. Our method supports incident responders or digital forensic investigators.

4.3 Overview

Figure 3 shows an overview of our method. First, our method constructs a corpus from known malicious and benign traffic. Each traffic stream is translated and separated by the previously mentioned method. In this paper, we use 2 reading methods. **Table 1** shows the contents of the reading methods. As a matter of course, the contents do not contain any raw FQDNs or IP addresses. Each method reads only each layer. First, our method reads summary lines of TShark. The number of the lines is fixed and predetermined. Next, the summary lines are converted into a paragraph. Our method then repeats this process to generate paragraphs from traffic. A paragraph does not contain duplicate summary lines.

In the training phase, the Doc2vec constructs a vector space from the corpus, and converts each paragraph into vectors with the labels. These labeled vectors are training data for classifiers. The classifiers are Support Vector Machine (SVM), Random Forests (RF), and Multi-Layer Perceptron (MLP). An SVM model is a representation of the training data as points in space, mapped so that the training data of the separate categories are divided by a clear gap that is as wide as possible. Test data are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. RF are an

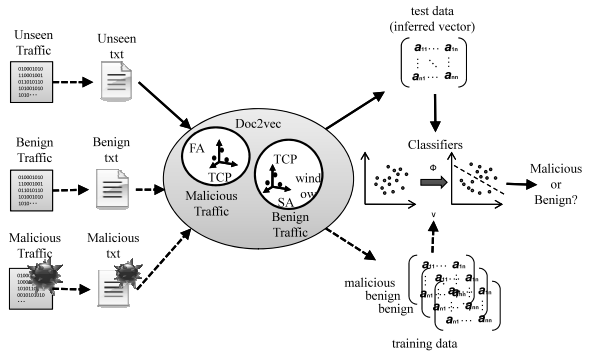

Fig. 3 An overview of the proposed method.

Table 1 A summary of the reading methods.

Method	Layer	Contents
Method 1	Transport Layer	protocol, port number, size
Method 2	Application Layer	FQDN, path, user agent

ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. MLP is a class of feedforward artificial neural network, which consists of at least three layers of nodes. Each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Test data are mapped into that same space and predicted to belong to a category based on which side of the gap they fall. Given a set of training data, each labeled as belonging to one or the other of two categories, this training algorithm builds a model that assigns new examples to one category or the other.

In the test phase, we convert new traffic into vectors. These unlabeled vectors are test data for the classifier. Then, we input these unlabeled vectors to the trained classifiers, and can obtain a predicted label. The predicted label is either malicious or benign.

4.4 Implementation

The proposed method was developed by Python-2.7 with open source machine learning libraries, gensim-1.01 [31] and scikit-learn-0.18.0 [32].

Gensim is a Python library to realize unsupervised semantic modelling from plain text, and includes a Doc2vec model. **Table 2** shows the parameters for the Doc2vec model. We set the dimensionality of the feature vectors 100, and chose DBoW which was the extension of skip-gram. The window is the maximum distance between the predicted word and context words used for prediction within a document. The minimum count is the threshold value to ignore all words with total frequency lower than this.

Scikit-learn is a machine-learning library for Python that provides tools for data mining with a focus on machine learning, and supports SVM and RF. Our method uses a SVC function with a linear kernel for SVM. Our method also uses a RandomForest-Classifier function for RF.

Chainer is a flexible Python framework for neural networks, which supports MLP with CUDA computation. We use CUDA 8.0 and cuDNN-6.0. **Table 3** shows the parameters for the MLP model. The number of input layer units is the dimensionality of

Table 2 The parameters for the Doc2vec model.

Dimensionality of the feature vectors	100
Window	15
Minimum count	2
Number of epochs	30
Training algorithm	DBoW

Table 3 The parameters for the MLP model.

Number of input layer units	100
Number of hidden layer units	500
Number of labels	2
Activation function	ReLU
Dropout ratio	0
Minibatch size	100
Optimizer	Adam

the test data. Thus, the dimensionality is 100 as we mentioned before. The number of labels is 2, namely benign or malicious. ReLU (Rectified Linear Unit) is an activation function defined as follows.

$$f(x) = x^+ = \max(0, x)$$

ReLU is also known as a ramp function and has been used in convolutional networks more effectively than the widely used logistic sigmoid. Adam (Adaptive moment estimation) is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments [33]. This method is well suited for problems that are enormous in terms of data and parameters, and also appropriate for non-stationary objectives and problems with very noisy and sparse gradients. We use cross entropy to define the loss function in optimization.

5. Experiment

5.1 Dataset

To reveal the effectiveness to malicious traffic, we use captured pcap files from actual malware between 2014 and 2017 (MTA dataset), which were downloaded from the website MALWARE-TRAFFIC-ANALYSIS.NET [6]. We also use the D3M (Drive-by Download Data by Marionette) dataset for the cross-dataset validation, and the NCD (Normal Communication Data in MWSCup 2014). These datasets are parts of MWS datasets [34], and include pcap files. The MTA and D3M contain malicious traffic and the NCD contains benign traffic. **Table 4** shows the sizes.

The MTA dataset contains malicious traffic from the following malware, Angler EK, Fiesta-EK, FlashPack-EK, Magnitude EK, Neutrino EK, Nuclear EK, and RIG EK. The D3M is a set of packet traces collected from the web-client, high-interaction honeypot system, which is based on Internet Explorer on Windows OS with several vulnerable plugins, such as Adobe Reader, Flash Player, Java, and so on. This dataset contains traffic from Blackhole EK, Elenore, and Mpack.

We extracted paragraphs from the MTA and D3M datasets with our method, and added the malicious labels to the paragraphs. We extracted paragraphs from the NCD datasets with our method, and added the benign labels to the paragraphs. This is based on

Table 4 The sizes of the datasets.

MTA		D3M		NCD	
year	size	year	size	year	size
2014	238 M	2010	130 M	2014	6.4 G
2015	186 M	2011	24.8 M	-	-
2016	373 M	2012	33.2 M	-	-
2017	109 M	2013	14.6 M	-	-
-	-	2014	23.3 M	-	-
-	-	2015	334 M	-	-

the assumption mentioned in Section 4.2. Note that the purpose of this experiment is not detecting malicious URLs. The purpose of this experiment is classifying 2 types of traffic. Then, we compound the malicious and benign paragraphs into a dataset. We split the dataset into training data and test data. The training data is a dataset used to build the final model. The test data is a dataset used to provide an unbiased evaluation of a final model fit on the training dataset. The proposed method uses only training data to construct a corpus because our method presumes that the test data is completely new traffic in practical environment. To confirm this assumption, we investigated unique indicators such as FQDN or IP address. **Table 5** shows the number of duplicated indicators between each year in the MTA dataset. Thus, the test data does not contain much duplicated indicators. Hence, it seems difficult to detect malicious traffic from the training data.

5.2 Metrics

In this experiment, we use 3 metrics: Precision (P), Recall (R), and F-measure (F). These metrics are used to evaluate the performance of classification tasks.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - measure = \frac{2Recall \times Precision}{Recall + Precision}$$

TP (True Positive), *FP* (False Positive), *TN* (True Negative) and *FN* (False Negative) are defined as shown in **Table 6**.

In this paper, malicious traffic is a paragraph which contains malicious summary lines and the context. Our method interprets multiple summary lines as a paragraph and adds the label to the paragraph. Hence, we calculate TP and FP from the number of classified paragraphs.

5.3 Fundamental Experiment

First, we attempt to reveal the effectiveness of Doc2vec. In this fundamental experiment, we compounded the malicious summary lines and benign summary lines into datasets at the same rate. As a baseline method, we also use a Bag-of-Words (BoW) model. BoW is a simplifying representation used in natural language processing. The most common type of features calculated from BoW is the number of times a term appears in the sentence. We conducted timeline analysis and cross-dataset validation. In the timeline analysis, we chose an annual traffic as training data, and subsequent traffic as test data. In the cross-dataset validation, we chose the MTA and D3M as training data respectively, and the other dataset is the test data. In this experiment, our method interprets 100 summary lines as a paragraph.

Table 5 The number of duplicated indicators between each year in the MTA dataset.

training data	test data	duplicated FQDNs	duplicated IP addresses
2014	2015	312 / 3172	204 /2870
2014	2016	55 / 1032	47 / 21041
2014	2017	21 /96	17 /5473
2015	2016	55 / 1032	69 / 21041
2015	2017	8 /96	7 /5473
2016	2017	12 /96	50 /5473

Table 6 Confusion matrix for two possible outcomes.

		True label	
		Positive	Negative
Predicted label	Positive	*TP*	*FP*
	Negative	*FN*	*TN*

Table 7 shows the results of the fundamental timeline analysis.

Each number of benign and malicious vectors is shown in parentheses. Contrary to expectations, BoW was generally more effective than Doc2vec with the method 1. Both F-measures maintain generally constants over three years, and the best one has reached 0.96. This result means that word frequency is the most distinctive feature in a transport layer. Doc2vec was generally more effective than BoW with the method 2. The best F-measure has reached 0.98 in the next year, and both F-measures gradually decrease. This result means that Doc2vec captures distinctive features other than word frequency in an application layer.

Table 8 shows the results of the cross-dataset validation.

Each number of benign and malicious vectors is shown in parentheses. BoW proved totally ineffective with method 1. Furthermore, even Doc2vec was not effective enough. Therefore, BoW was totally ineffective under other environments. Besides, it is difficult to classify new traffic from transport layer information under other environments.

Doc2vec was generally more effective than BoW with the method 2. In the case that we used the MTA for training data, the best F-measure has reached 0.97. This result means that the MTA is superior to D3M as a training data. Moreover, using Doc2vec is effective for classifying new traffic from application layer information under other environments.

As a result, using Doc2vec in an application layer was the most effective. In the timeline analysis, the best F-measure achieves 0.98. The F-measure maintains 0.80 for 3 years. In the cross-dataset validation, the best F-measure achieves 0.97.

5.4 Practical Experiment

In the fundamental experiment, we concluded that using Doc2vec in an application layer was the most effective. Next, we attempt to reveal the performance of our method in a practical environment and use several classifiers. In actual network traffic, most traffic is benign. Therefore, we compounded the malicious summary lines and all the benign summary lines into imbalanced datasets. In these datasets, benign traffic is dominant as with actual network traffic. We split the datasets into training data

Table 7 The result of the fundamental timeline analysis.

method	training data	test data	model	NCD (Benign)			MTA (Malicious)		
				P	R	F	P	R	F
1	2014 (487)	2015 (352)	BoW	0.97	0.93	0.95	0.93	0.97	0.95
			Doc2vec	0.94	0.80	0.86	0.93	0.95	0.88
		2016 (687)	BoW	1.00	0.92	0.96	0.92	1.00	0.96
			Doc2vec	0.97	0.78	0.87	0.82	0.97	0.89
		2017 (162)	BoW	0.99	0.92	0.95	0.92	0.99	0.96
			Doc2vec	0.99	0.80	0.89	0.83	1.00	0.91
2	2014 (1,262)	2015 (926)	BoW	0.95	0.80	0.87	0.83	0.96	0.89
			Doc2vec	0.99	0.98	0.98	0.98	0.99	0.98
		2016 (1,560)	BoW	0.88	0.83	0.85	0.84	0.89	0.86
			Doc2vec	0.88	0.97	0.92	0.96	0.87	0.92
		2017 (437)	BoW	0.84	0.81	0.83	0.83	0.85	0.84
			Doc2vec	0.77	0.98	0.86	0.96	0.69	0.80

Table 8 The result of the fundamental cross-dataset validation.

method	training data	test data	model	NCD (Benign)			Malicious		
				P	R	F	P	R	F
1	MTA (1,693)	D3M (840)	BoW	0.57	0.96	0.71	0.90	0.36	0.52
			Doc2vec	0.65	0.89	0.75	0.86	0.59	0.70
	D3M (840)	MTA (1,693)	BoW	0.40	0.96	0.56	0.93	0.25	0.40
			Doc2vec	0.59	0.93	0.72	0.95	0.66	0.78
2	MTA (4,196)	D3M (2,689)	BoW	0.89	0.88	0.88	0.87	0.89	0.88
			Doc2vec	0.98	0.96	0.97	0.96	0.98	0.97
	D3M (2,689)	MTA (4,196)	BoW	0.74	0.98	0.84	0.97	0.67	0.79
			Doc2vec	0.79	0.99	0.88	0.99	0.74	0.85

and test data to conduct 10-fold cross-validation, timeline analysis and cross-dataset validation. In the 10-fold cross-validation, we use all the datasets. In the timeline analysis and cross-dataset validation, we switch malicious datasets for the training data and test data. We use the first half of the NCD dataset as the training data and the remaining half as the test data. In this experiment, our method interprets 50 summary lines as a paragraph. This parameter was decided based on an empirical approach.

Table 9 shows the results of the 10-fold cross-validation.

Each number of malicious vectors is shown in parentheses. Each number of the benign and malicious vectors is 100,864 and 36,755 respectively. RF is less accurate than other classifiers. The other classifiers maintain good accuracy in a practical environment.

Table 10 shows the results of the practical timeline analysis.

Malicious vector counts are shown in parentheses. The training and test benign vectors were respectively 100,864 and 100,495. When comparing malicious vectors with benign ones, the maximum number of benign vectors are approximately a 38-fold increase from that of malicious vectors. Therefore, these datasets are imbalanced as we expected. SVM and MLP achieved good accuracy on average. The best F-measure has reached 0.82 in the next year, and the both F-measures gradually decrease. RF is less accurate and more unstable than SVM and MLP. Each F-measure decreases gradually with time.

Table 11 shows the runtime of the practical timeline analysis.

In the timeline analysis, our method required 30 minutes on average. The total runtime includes the time for making a dataset, training a classifier and prediction. Our method can train classi-

Table 9 The result of the 10-fold cross-validation.

classifier	Benign			Malicious		
	P	R	F	P	R	F
SVM	0.99	0.99	0.99	0.96	0.96	0.96
RF	0.92	0.98	0.95	0.94	0.77	0.84
MLP	0.98	0.98	0.98	0.96	0.95	0.95

Table 11 The runtime of the practical timeline analysis.

training data	test data	classifier	total runtime	prediction time
2014	2015	SVM	28 m	10.2 s
		RF	24 m	0.1 s
		MLP	27 m	2.9 s
2014	2016	SVM	30 m	11.8 s
		RF	25 m	0.2 s
		MLP	31 m	3.2 s
2014	2017	SVM	28 m	9.9 s
		RF	25 m	0.2 s
		MLP	30 m	2.9 s
2015	2016	SVM	27 m	8.8 s
		RF	25 m	0.2 s
		MLP	31 m	3.7 s
2015	2017	SVM	27 m	7.9 s
		RF	25 m	0.2 s
		MLP	30 m	3.5 s
2016	2017	SVM	56 m	26.0 s
		RF	25 m	0.2 s
		MLP	31 m	3.1 s

Table 10 The result of the practical timeline analysis.

training data	test data	classifier	Benign			Malicious		
			P	R	F	P	R	F
2014 (4,028)	2015 (2,639)	SVM	0.99	0.98	0.99	0.73	0.87	0.80
		RF	0.96	1.00	0.98	0.91	0.31	0.47
		MLP	0.99	0.98	0.99	0.79	0.86	0.82
2014 (4,028)	2016 (16,737)	SVM	0.95	0.98	0.96	0.82	0.67	0.74
		RF	0.90	1.00	0.95	0.95	0.33	0.50
		MLP	0.95	0.98	0.97	0.87	0.68	0.76
2014 (4,028)	2017 (3,615)	SVM	1.00	0.97	0.98	0.55	0.91	0.68
		RF	0.98	1.00	0.99	0.88	0.46	0.60
		MLP	1.00	0.98	0.99	0.63	0.92	0.75
2015 (2,639)	2016 (16,737)	SVM	0.95	0.98	0.97	0.87	0.69	0.77
		RF	0.88	1.00	0.94	0.99	0.18	0.31
		MLP	0.95	0.99	0.97	0.95	0.69	0.80
2015 (2,639)	2017 (3,615)	SVM	0.99	0.98	0.99	0.61	0.68	0.65
		RF	0.97	1.00	0.98	0.89	0.49	0.09
		MLP	0.99	0.99	0.99	0.78	0.66	0.72
2016 (16,737)	2017 (3,615)	SVM	1.00	0.98	0.99	0.66	0.94	0.78
		RF	0.98	0.99	0.98	0.58	0.49	0.53
		MLP	1.00	0.99	0.99	0.71	0.93	0.80

Table 12 The result of the practical cross-dataset validation.

training data	test data	classifier	Benign			Malicious		
			P	R	F	P	R	F
MTA (36,755)	D3M (19,620)	SVM	0.91	0.96	0.94	0.73	0.53	0.62
		RF	0.89	0.97	0.93	0.71	0.39	0.50
		MLP	0.94	0.97	0.95	0.80	0.67	0.73
D3M (19,620)	MTA (36,755)	SVM	0.79	0.98	0.88	0.93	0.27	0.42
		RF	0.74	1.00	0.85	0.88	0.04	0.08
		MLP	0.78	0.99	0.87	0.93	0.21	0.35

fiers in advance. Therefore, we focus on the prediction time. At the expense of accuracy, RF is the fastest. SVM requires much more time than other classifiers. Thus, MLP is the best in terms of the balance between the accuracy and runtime.

Table 12 shows the results of the practical cross-dataset validation.

Each number of malicious vectors is shown in parentheses. Each number of the train and test benign vectors is 100,864 and 100,495 respectively. In the case that we used the MTA for training data, the best F-measure has reached 0.73. This result means that the MTA is superior to the D3M as a training data, and is not inconsistent with the previous result. RF is less accurate than the other classifiers.

6. Discussion

6.1 Mechanism

In a transport layer, word frequency was the most distinctive feature. The most frequent words were traffic sizes. In a sense, this is rote memorization. In fact, BoW was totally ineffective in other environments. Furthermore, even Doc2vec was not effective enough. Therefore, we concluded that it was difficult to classify new traffic from transport layer information.

In an application layer, Doc2vec captured distinctive features other than word frequency. In fact, Doc2vec was also effective under other environments. Needless to say, word frequency is a fundamental element in a linguistic approach. However, the unique word count in network traffic is unrestricted. Representing unrestricted traffic with only word frequency has serious limitations. Doc2vec is based on the distributional hypothesis that words occurring in similar context tend to have similar meanings. Doc2vec allows calculating the semantic similarity between two traffic streams and infer similar traffic semantically. Hence, network traffic in an application layer might have the necessary context.

6.2 Accuracy

In the fundamental experiment, using Doc2vec in an application layer was the most effective. In the timeline analysis, the best F-measure achieved 0.98. In the cross-dataset validation, the best F-measure achieved 0.97. In actual network traffic, however, most traffic is benign.

Therefore, we conducted the practical experiment with imbalanced datasets which simulate actual network traffic. In the timeline analysis, the best F-measure achieved 0.82. In the cross-dataset validation, the best F-measure achieved 0.73. The best classifier was MLP in terms of the balance between the accuracy and runtime.

6.3 Comparison

In terms of accuracy, our method has slightly lower accuracy

than the previous method [5]. In regard to other previous methods, it is not feasible to compare under fair conditions since many previous works use their own datasets which are not open to the public. One possible reason is that the benign datasets may contain sensitive information. Moreover, some previous methods were evaluated with a balanced dataset, which is not practical. This paper provides better reproducibility, because our method is evaluated with public datasets. Hence, we attempt a qualitative comparison.

Our method examines a variety of malicious traffic in the same method. All we have to do is input malicious and benign pcap files. Our method attempts to detect malicious traffic regardless of the attack techniques. No prior knowledge of the attack techniques is required. The previous method [5] has the same features. However, the previous method supports only http traffic. Our method is available in any protocols which TShark can analyze in an application layer. If attackers change the attack techniques or protocols, our methods expect to learn the features automatically. Besides our method does not require devising feature vectors. However, other approaches with NN [25], [26], [27] require basic feature vectors. Hence, our method is adaptable to many attack techniques.

6.4 Limitation

Our method has some limitations. The most major limitation is based on the assumption described in Section 4.2. In this paper, we assume that several benign summary lines are related to the malicious summary lines. Our method examines a paragraph comprehensively and determines whether a paragraph is malicious or not. This means our method does not detect only malicious summary lines. In actuality, more benign summary lines might be mixed into a malicious paragraph. Hence, the detection rate of our method will probably be affected by the mixed benign summary lines. In practical use, we assume our method monitors internal network traffic on a network switch, which can monitor all internal IP addresses. Hence, we can use internal IP addresses to separate each traffic streams. This will mitigate the effect of the mixture.

By the same assumption, operators have to find malicious lines from the detected malicious paragraph. To extract malicious lines, we can use traditional signatures or reputation-based methods such as whois database. Our method expects that the operators perceive intrusion based on the result. In this paper, the operators contain not only network operators, but also incident responders or digital forensic investigators. As a matter of course, it is difficult to detect malicious traffic in APT attacks. Hence, we believe even the awareness is important and useful for incident responders or digital forensic investigators.

Another limitation is based on the comprehensiveness of the datasets. In this paper, we used datasets consisting of DbD attacks and C&C traffic. In these datasets, it is difficult to decide whether a summary line is benign or malicious independently. Hence, appropriate labeled datasets are required. In this experiment, we determined several parameters based on an empirical approach. To determine the optimum parameters, our method might require more large-scale datasets. However, due to the dataset constraints, using large scale dataset lies outside the scope of this paper.

7. Conclusion

In this paper, we extend the previous method to a generic detection method which supports any protocol. This paper demonstrates the performance with captured traffic which is downloaded from the website MALWARE-TRAFFIC-ANALYSIS.NET. In the fundamental experiment, the best F-measure achieves 0.98 in the timeline analysis and 0.97 in the cross-dataset validation. As a result, using Doc2vec in an application layer is effective for classifying new traffic. Furthermore, this paper generates imbalanced datasets which simulate actual network traffic, and verifies the performance of the proposed method in practical environment. In the practical experiment, the best F-measure achieves 0.82 in the timeline analysis and 0.73 in the cross-dataset validation.

Applying our method to more large-scale datasets is a topic for future study. A large network has many clients which access various websites. In terms of many clients, we can use internal IP addresses to separate the traffic. Nevertheless, we believe evaluating the impact of the mixture rate is also beneficial. Regarding various websites, benign traffic contains more words and might occupy feature representation. One improvement plan is using other NLP techniques to select the important words. These techniques enable reducing various words in dominant benign traffic. Thus, we can generate a balanced corpus with the same number of words from each traffic. Large-scale datasets are also required to determine the optimum parameters. In this paper, we had to determine several parameters based on an empirical approach due to the dataset constraints. Determining the optimum parameter under any environment is a topic for future study.

References

[1] Jodavi, M., Abadi, M. and Parhizkar, E.: DbDHunter: An ensemble-based anomaly detection approach to detect drive-by download attacks, *2015 5th International Conference on Computer and Knowledge Engineering* (*ICCKE*), pp.273–278 (online), DOI: 10.1109/ICCKE.2015.7365841 (2015).

[2] Gu, G., Perdisci, R., Zhang, J. and Lee, W.: BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection, *Proc. 17th USENIX Security Symposium*, pp.139–154 (2008) (online), available from ⟨http://www.usenix.org/events/sec08/tech/full_papers/gu/gu.pdf⟩.

[3] Bilge, L., Balzarotti, D., Robertson, W.K., Kirda, E. and Kruegel, C.: Disclosure: Detecting botnet command and control servers through large-scale NetFlow analysis, *28th Annual Computer Security Applications Conference, ACSAC 2012*, pp.129–138 (online), DOI: 10.1145/2420950.2420969 (2012).

[4] Mimura, M., Otsubo, Y., Tanaka, H. and Tanaka, H.: A Practical Experiment of the HTTP-Based RAT Detection Method in Proxy Server Logs, *12th Asia Joint Conference on Information Security, AsiaJCIS 2017*, pp.31–37 (online), DOI: 10.1109/AsiaJCIS.2017.13 (2017).

[5] Mimura, M. and Tanaka, H.: Leaving All Proxy Server Logs to Paragraph Vector, *JIP*, Vol.26, pp.804–812 (online), DOI: 10.2197/ipsjjip.26.804 (2018).

[6] MALWARE-TRAFFIC-ANALYSIS.NET, available from ⟨http://www.malware-traffic-analysis.net/⟩.

[7] Mimura, M. and Tanaka, H.: Reading Network Packets as a Natural Language for Intrusion Detection, *Information Security and Cryptology - ICISC 2017 - 20th International Conference*, pp.339–350 (online), DOI: 10.1007/978-3-319-78556-1_19 (2017).

[8] Wang, K. and Stolfo, S.J.: Anomalous Payload-Based Network Intrusion Detection, *Proc. Recent Advances in Intrusion Detection: 7th International Symposium, RAID 2004*, pp.203–222 (online), DOI: 10.1007/978-3-540-30143-1_11 (2004).

[9] Moore, D., Shannon, C., Brown, D.J., Voelker, G.M. and Savage, S.:

Inferring Internet denial-of-service activity, *ACM Trans. Comput. Syst.*, Vol.24, No.2, pp.115–139 (online), DOI: 10.1145/1132026.1132027 (2006).

[10] Bailey, M., Oberheide, J., Andersen, J., Mao, Z.M., Jahanian, F. and Nazario, J.: Automated Classification and Analysis of Internet Malware, *Proc. Recent Advances in Intrusion Detection, 10th International Symposium, RAID 2007*, pp.178–197 (online), DOI: 10.1007/978-3-540-74320-0_10 (2007).

[11] Song, H. and Turner, J.S.: Toward Advocacy-Free Evaluation of Packet Classification Algorithms, *IEEE Trans. Computers*, Vol.60, No.5, pp.723–733 (online), DOI: 10.1109/TC.2010.252 (2011).

[12] Karagiannis, T., Papagiannaki, K. and Faloutsos, M.: BLINC: multilevel traffic classification in the dark, *Proc. ACM SIGCOMM 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp.229–240 (online), DOI: 10.1145/1080091.1080119 (2005).

[13] Antonakakis, M., Perdisci, R., Dagon, D., Lee, W. and Feamster, N.: Building a Dynamic Reputation System for DNS, *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pp.273–290 (online), available from ⟨http://www.usenix.org/events/sec10/tech/full_papers/Antonakakis.pdf⟩ (2010).

[14] Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou, N. and Dagon, D.: Detecting Malware Domains at the Upper DNS Hierarchy, *Proc. 20th USENIX Security Symposium* (online), available from ⟨http://static.usenix.org/events/sec11/tech/full_papers/Antonakakis.pdf⟩ (2011).

[15] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W. and Dagon, D.: From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware, *Proc. 21st USENIX Security Symposium*, pp.491–506 (online), available from ⟨https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/antonakakis⟩ (2012).

[16] Rahbarinia, B., Perdisci, R. and Antonakakis, M.: Segugio: Efficient Behavior-Based Tracking of Malware-Control Domains in Large ISP Networks, *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp.403–414 (online), DOI: 10.1109/DSN.2015.35 (2015).

[17] Krügel, C. and Vigna, G.: Anomaly detection of web-based attacks, *Proc. 10th ACM Conference on Computer and Communications Security, CCS 2003*, pp.251–261 (online), DOI: 10.1145/948109.948144 (2003).

[18] Choi, H., Zhu, B.B. and Lee, H.: Detecting Malicious Web Links and Identifying Their Attack Types, *2nd USENIX Conference on Web Application Development, WebApps'11* (2011) (online), available from ⟨https://www.usenix.org/conference/webapps11/detecting-malicious-web-links-and-identifying-their-attack-types⟩.

[19] Ma, J., Saul, L.K., Savage, S. and Voelker, G.M.: Learning to detect malicious URLs, *ACM TIST*, Vol.2, No.3, pp.30:1–30:24 (online), DOI: 10.1145/1961189.1961202 (2011).

[20] Zhao, P. and Hoi, S.C.H.: Cost-sensitive online active learning with application to malicious URL detection, *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013*, pp.919–927 (online), DOI: 10.1145/2487575.2487647 (2013).

[21] Invernizzi, L., Miskovic, S., Torres, R., Kruegel, C., Saha, S., Vigna, G., Lee, S. and Mellia, M.: Nazca: Detecting Malware Distribution in Large-Scale Networks, *21st Annual Network and Distributed System Security Symposium, NDSS 2014* (2014) (online), available from ⟨https://www.ndss-symposium.org/ndss2014/nazca-detecting-malware-distribution-large-scale-networks⟩.

[22] Nelms, T., Perdisci, R., Antonakakis, M. and Ahamad, M.: WebWitness: Investigating, Categorizing, and Mitigating Malware Download Paths, *24th USENIX Security Symposium, USENIX Security 15*, Jung, J. and Holz, T. (Eds.), USENIX Association, pp.1025–1040 (2015).

[23] Bartos, K., Sofka, M. and Franc, V.: Optimized Invariant Representation of Network Traffic for Detecting Unseen Malware Variants, *25th USENIX Security Symposium, USENIX Security 16*, Holz, T. and Savage, S. (Eds.), USENIX Association, pp.807–822 (2016).

[24] Mimura, M. and Tanaka, H.: Heavy Log Reader: Learning the Context of Cyber Attacks Automatically with Paragraph Vector, *Proc. Information Systems Security - 13th International Conference, ICISS 2017*, pp.146–163 (online), DOI: 10.1007/978-3-319-72598-7_9 (2017).

[25] Potluri, S., Henry, N.F. and Diedrich, C.: Evaluation of hybrid deep learning techniques for ensuring security in networked control systems, *ETFA*, pp.1–8, IEEE (online), available from ⟨http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8233358⟩ (2017).

[26] Alom, M.Z. and Taha, T.M.: Network intrusion detection for cyber security using unsupervised deep learning approaches, *2017 IEEE National Aerospace and Electronics Conference* (*NAECON*), pp.63–69 (online), DOI: 10.1109/NAECON.2017.8268746 (2017).

[27] Wang, Z.: Deep Learning-Based Intrusion Detection With Adversaries, *IEEE Access*, Vol.6, pp.38367–38384 (2018).

[28] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J.: Distributed Representations of Words and Phrases and their Compositionality, *Proc. Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Systems 2013*, pp.3111–3119 (online), available from ⟨http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality⟩ (2013).

[29] Le, Q.V. and Mikolov, T.: Distributed Representations of Sentences and Documents, *Proc. 31st International Conference on Machine Learning, ICML 2014*, pp.1188–1196 (2014) (online), available from ⟨http://jmlr.org/proceedings/papers/v32/le14.html⟩.

[30] tshark - Dump and analyze network traffic, available from ⟨https://www.wireshark.org/docs/man-pages/tshark.html⟩.

[31] gensim, available from ⟨https://radimrehurek.com/gensim/⟩.

[32] scikit-learn, available from ⟨http://scikit-learn.org/⟩.

[33] Kingma, D. and Ba, J.: Adam: A Method for Stochastic Optimization, *International Conference on Learning Representations* (2014).

[34] Hatada, M., Akiyama, M., Matsuki, T. and Kasama, T.: Empowering Anti-malware Research in Japan by Sharing the MWS Datasets, *JIP*, Vol.23, No.5, pp.579–588 (online), DOI: 10.2197/ipsjjip.23.579 (2015).

Mamoru Mimura received his B.E. and M.E. in Engineering from National Defense Academy of Japan, in 2001 and 2008 respectively. He received his Ph.D. in Informatics from the Institute of Information Security in 2011 and M.B.A. from Hosei University in 2014. During 2001–2017, he was a member of the Japanese Maritime Self-Defense Force. During 2011–2013, he was with the National Information Security Center. Since 2014, he has been a researcher in the Institute of Information Security. Since 2015, he has been with the National center of Incident readiness and Strategy for Cybersecurity. Currently, he is an Associate Professor in the Department of C.S., National Defense Academy of Japan.