Malicious PDF Detection Scheme Using the Useful Feature Based on Non-Frequent Keywords in a File

Hiroya Kato, Shuichiro Haruta, and Iwao Sasase
Dept. of Information and Computer Science, Keio University
3-14-1 Hiyoshi, Kohoku, Yokohama, Kanagawa 223-8522, Japan,
Email: kato@sasase.ics.keio.ac.jp

***Abstract*—Detecting malicious PDFs (Portable Document Format) is imperative. As a malicious PDF detection scheme, we focus on the scheme leveraging the fact that the frequency of internal components called keywords is different between legitimate and malicious PDFs. That scheme uses the keywords which frequently appear in the dataset to detect malicious PDFs. However, the keywords appeared only in legitimate or malicious PDFs can be ignored in the conventional scheme. In ignored keywords, if there exist the keywords which can have useful features, that scheme cannot detect malicious PDFs which possess such keywords. In this paper, we propose malicious PDF detection scheme using the useful feature based on non-frequent keywords in a file. Thus, in order to evaluate such keywords precisely, we utilize c_{sub} which represents the deference of keywords appeared only in legitimate and malicious dataset. Furthermore, we use n_{keyword} which denotes the number of non-duplicate keywords appeared in a file. In this way, we can evaluate keywords that the conventional scheme ignores. In order to prevent c_{sub} and n_{keyword} from degrading the detection performance, we get the feature which quantitatively represents maliciousness of a PDF by applying fuzzy inference to these features. Our scheme utilizes this feature with conventional scheme's features to detect malicious PDFs. By computer simulation with real dataset, we demonstrate our scheme can reduce both false positive and false negative.**

***Index Terms*—Malicious PDF detection, Machine Learning, Fuzzy Inference, JavaScript**

I. Introduction

Recently, PDF (Portable Document Format) is widespread as one of the most standard document formats all over the world [1]. Since PDF has flexible document format, various contents such as hyperlinks or images can be embedded. On the other hand, its inner structure is complex and general users are unaware of its structure in detail. Unfortunately, attackers abuse this complex structure and embed malwares into the PDF. In many case, JavaScript is used to attack and it is performed by exploiting the vulnerability of Adobe Acrobat Reader [2]. In worst cases, the attack is succeeded when user just opens the PDF. To make matters worse, such malicious PDFs are sent to users by e-mail [3]. Since many people are familiar with PDFs, they tend to open them without caution. Therefore, even if attached PDF is malicious, receivers are likely to open them easily. In order to prevent these dangerous attacks, malicious PDF detection is imperative.

As a malicious PDF detection scheme, Maiorca et al. [4] propose the scheme leveraging the fact that the frequency of internal components called keywords is different between legitimate and malicious PDFs. And then, it detects malicious PDF by using machine learning. That scheme selects keywords which frequently appear in the dataset and makes feature vectors for each file by using the number of their occurrence.

However, in the ignored keywords, the keywords whose number of occurrence is closed to threshold values may have useful features to detect malicious PDF.

In order to evaluate such keywords precisely, in this paper we propose malicious PDF detection scheme using the useful feature based on non-frequent keywords in a file. We utilize c_{sub} which represents the deference of keywords appeared only in legitimate and malicious datasets. By doing this, we can mainly focus on the keywords which are ignored in the conventional scheme. Furthermore, we discover the number of keywords used in legitimate and malicious PDFs is different. Thus, we use n_{keyword} which denotes the number of non-duplicate keywords appeared in a file. However, if these features are individually utilized for training of machine learning, there exist the situations where the judgement whether malicious or not is unclear. Therefore, the detection performance can be degraded by these features. In order to prevent this bad effect, we get a score which quantitatively represents maliciousness of a PDF by applying fuzzy inference [5] to these features. We utilize this feature with conventional scheme ' s features to detect malicious PDFs.

The rest of this paper is constructed as follows. In Section II, we explain about PDF's internal structure. In Section III, we summarize related works and describe the conventional scheme. In Section IV, the proposed scheme is presented in detail. In Section V, we evaluate our scheme by the computer simulation with real dataset. Finally, we conclude this paper in Section VI.

II. PDF's internal structure

PDFs have the component called "objects" and PDF viewers repeatedly refer to them and display PDF. At first, PDF viewers refer to the root object and confirm the next objects. This referred objects are called indirect objects and they are used to display characters or images. PDFs mainly consist of four components which are header, body, cross-reference table, and trailer [6] [7]. The version information is written in header. Body is the main part of the PDF file and contains all the PDF objects to display characters or images and so on. Malicious JavaScript code and malware are also embedded in the body. Cross-reference table indicate the position of all indirect objects and trailer gives the location of the root object. The most important internal component of PDF is the string called ”keywords” which begins with slash ”/” and it is used to determine the actions of PDF. For example, the keyword /JavaScript is included in files to embed JavaScript in PDF.

III. Related work and conventional scheme

A. Related work

In order to detect malicious PDF, two representative schemes have been proposed [4] [8]. Liu et al. propose the scheme which analyzes JavaScript embedded in PDF [8]. Since the malicious PDFs often include malicious JavaScript codes, they can detect such PDF as malicious. However, since it is difficult to analyze the code in the case where the code is compressed or encrypted, that scheme cannot detect malicious PDF. On the other hand, Maiorca et al. propose the scheme which focuses on internal components of PDF [4]. That scheme leverages the fact that the frequency of keywords used in legitimate and malicious PDFs is different. They detect malicious PDF by using machine learning.

Although above two malicious PDF detection schemes are proposed, we pay attention to [4] because that scheme can detect malicious PDF even if analyzing JavaScript code is difficult. We explain the scheme [4] as the conventional scheme in the next section.

B. Conventional scheme

The main idea of the conventional scheme is leveraging the fact that the frequency of keywords used in legitimate and malicious PDFs is different. In the legitimate PDF, there are many keywords for displaying characters or pictures. On the other hand, in the malicious PDF, since it is not necessary to display them, the number of such keywords are relatively small. In order to obtain useful keywords which hardly depend on training dataset, each keyword is classified into two classes by K-Means clustering algorithm [9], where $K = 2$. Finally, that scheme selects keywords of the upper class and makes feature vectors for each file by using the number of their occurrence. After that, these feature vectors are fed into machine learning classifier such as Ada boost [10] and it detects malicious PDF.

1) Keywords Selection:

Let D_{train} denote the training dataset and it includes $D_{\text{legitimate}}$ and $D_{\text{malicious}}$ which indicate the sets of legitimate and malicious PDFs respectively (i.e $D_{\text{train}} = D_{\text{legitimate}} \cup D_{\text{malicious}}$).

1) Creating two keywords' sets $K_{\text{legitimate}} = \{k_{\text{i}}|1 \leq i \leq n_{\text{legitimate}}\}$ and $K_{\text{malicious}} = \{k_{\text{j}}|1 \leq j \leq n_{\text{malicious}}\}$ from $D_{\text{legitimate}}$ and $D_{\text{malicious}}$, respectively. Here, $n_{\text{legitimate}}$ and $n_{\text{malicious}}$ are the numbers of non-duplicate keywords in $D_{\text{legitimate}}$ and $D_{\text{malicious}}$, respectively.
2) Let n_{k_i} denote the number of files which include k_{i} in $D_{\text{legitimate}}$. Similarly, let $n_{k_{\text{j}}}$ denote the number of files which include k_{j} in $D_{\text{malicious}}$. Deviding $K_{\text{legitimate}}$ into two classes by applying K-Means clustering algorithm to $n_{k_{\text{i}}}$ and the threshold value which is a boundary of two classes is determined. Furthermore, selecting the class that includes keywords which appear more than the threshold value as $K^{'}_{\text{legitimate}}$. The same procedure is executed to $K_{\text{malicious}}$, and $K^{'}_{\text{malicious}}$ is finaly obtained.
3) The keywords used as feature are the elements in $K_{\text{feature}} = K^{'}_{\text{legitimate}} \cup K^{'}_{\text{malicious}}$.

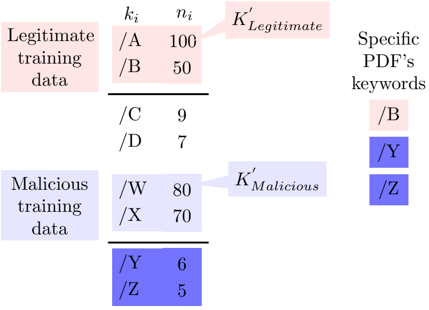


Fig. 1: Keywords which are not considered as a features

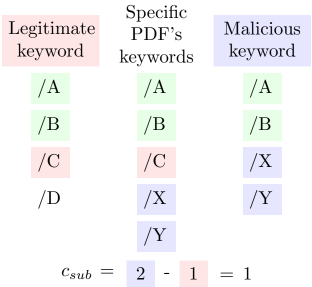


Fig. 2: An example of calculating c_{sub}

2) Shortcoming of the Conventional Scheme:

Since the keywords considered on the conventional scheme are selected from the keywords which frequently appear in the dataset, the others are ignored. However, in the ignored keywords, the keywords whose number of occurrence is closed to threshold values can be useful features to detect malicious PDF. Fig. 1 shows an example of this situation. In this figure, a malicious PDF which possesses keywords /B, /Y and /Z exists. Since keywords /Y and /Z appear only in malicious training dataset, they can be useful features to detect malicious PDF. However, because they are not selected as features, the conventional scheme cannot detect this malicious PDF.

IV. Proposed Scheme

We argue that most of keywords selected as features in the conventional scheme appear in both legitimate and malicious PDFs. Therefore, the keywords appeared only in legitimate or malicious can be ignored in the conventional scheme. In order to evaluate such keywords precisely, we utilize c_{sub} which represents the deference of keywords appeared only in legitimate and malicious dataset. Fig. 2 shows an example of calculating c_{sub}. As we can see from this figure, since the keywords appeared in both legitimate and malicious PDFs are

TABLE I: The average number of non-duplicate keywords appeared in a file

PDF	Average number of non-duplicate keywords
Legitimate PDF	102
Malicious PDF	34

canceled each other, we can mainly focus on the keywords which are ignored in the conventional scheme. Furthermore, we use n_{keyword} which denotes the number of non-duplicate keywords appeared in a file. This is because it tends to be different between legitimate and malicious as we can see from TABLE I. However, if these features are individually utilized for training of machine learning, there exist the situations where the judgement whether malicious or not is unclear. For example, we suppose that a malicious PDF which possesses $c_{\mathrm{sub}} = 2$ and $n_{\mathrm{keyword}} = 100$. In this case, because the number of keywords appeared in malicious is more than that of legitimate, c_{sub} indicates malicious feature. However, since the number of non-duplicate keywords is close to the average value of legitimate PDF, n_{keyword} indicates legitimate feature. Therefore, the detection performance can be degraded by these features. In order to prevent this bad effect, we apply fuzzy inference [5] to these features. Since fuzzy inference is the technique which expresses the ambiguous propositions, we can get a score which quantitatively represents maliciousness of a PDF. This score takes the value between 0 and 1. When the score is 1, it indicates that the PDF is malicious. We utilizes this feature with the conventional scheme's features to detect malicious PDFs.

A. Algorithm

1) Obtaining c_{sub} and n_{keyword}:

From all keywords included in a PDF file f, we count the number of keywords appeared in $K_{\mathrm{legitimate}}$ and call this $c_{\mathrm{legitimate}}$. Similarly, $c_{\mathrm{malicious}}$ is counted and c_{sub} is calculated as

$$c_{\mathrm{sub}} = c_{\mathrm{malicious}} - c_{\mathrm{legitimate}}. \tag{1}$$

Moreover, we count n_{keyword} which represents the number of non-duplicate keywords appeared in a file.

2) Definition of proposition about PDF:

In order to apply fuzzy inference to c_{sub} and n_{keyword}, four propositions are defined as follows:

1) If c_{sub} is large and n_{keyword} is also large, the PDF is malicious.
2) If c_{sub} is large and n_{keyword} is small, the PDF is malicious.
3) If c_{sub} is small and n_{keyword} is large, the PDF is legitimate.
4) If c_{sub} is small and n_{keyword} is also small, the PDF is neutral.

The score obtained by fuzzy inference is calculated based on these propositions.

3) Determination of membership function:

In order to quantitatively represent four propositions, membership functions are defined. These functions take the value between 0 and 1. When the value is close to 1, the proposition is close to truth and vice versa. Since it is difficult to define the parameters of membership functions in advance, they are determined by some experiments in Section V. The membership function $\mu_A(.)$ which quantitatively represents the condition ”c_{sub} is large” in the previous Section is defined as the following equation:

$$\mu_A(c_{\mathrm{sub}}) = \begin{cases} 0 & (c_{\mathrm{sub}} \leq a), \\ \frac{1}{b-a}(c_{\mathrm{sub}} - a) & (a < c_{\mathrm{sub}} < b), \\ 1 & (c_{\mathrm{sub}} \geq b), \end{cases} \tag{2}$$

where a and b are parameters of membership functions. Similarly, the membership function $\mu_A^{'}(.)$ which quantitatively expresses the condition ”c_{sub} is small” in the previous Section is also defined as the following equation:

$$\mu_A^{'}(c_{\mathrm{sub}}) = \begin{cases} 1 & (c_{\mathrm{sub}} \leq a), \\ -\frac{1}{b-a}(c_{\mathrm{sub}} - b) & (a < c_{\mathrm{sub}} < b), \\ 0 & (c_{\mathrm{sub}} \geq b). \end{cases} \tag{3}$$

Fig. 3 shows the shape of $\mu_A(.)$ and $\mu_A^{'}(.)$. As we can see from this figure, since $\mu_A(.)$ is the membership function which represents ”c_{sub} is large”, the larger c_{sub} takes the value, the closer $\mu_A(c_{\mathrm{sub}})$ gets to 1. Meanwhile, because $\mu_A^{'}(.)$ is the membership function which represents ”c_{sub} is small”, the larger c_{sub} takes the value, the closer $\mu_A^{'}(c_{\mathrm{sub}})$ gets to 0.

The membership function $\mu_B(.)$ which quantitatively expresses the condition ”n_{keyword} is large” in the previous Section is defined as the following equation:

$$\mu_B(n_{\mathrm{keyword}}) = \begin{cases} 0 & (n_{\mathrm{keyword}} \leq c), \\ \frac{1}{d-c}(n_{\mathrm{keyword}} - c) & (c < n_{\mathrm{keyword}} < d), \\ 1 & (n_{\mathrm{keyword}} \geq d), \end{cases} \tag{4}$$

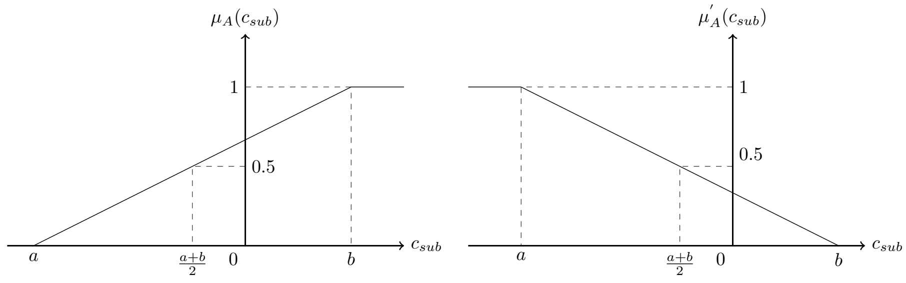
where c and d are parameters of membership functions. Similarly, the membership function $\mu_B^{'}(.)$ which quantitatively represents the condition ”n_{keyword} is small” in the previous Section is also defined as the following equation:

$$\mu_B^{'}(n_{\mathrm{keyword}}) = \begin{cases} 1 & (n_{\mathrm{keyword}} \leq c), \\ -\frac{1}{d-c}(n_{\mathrm{keyword}} - d) & (c < n_{\mathrm{keyword}} < d), \\ 0 & (n_{\mathrm{keyword}} \geq d). \end{cases} \tag{5}$$

Fig. 4 shows the shape of $\mu_B(.)$ and $\mu_B^{'}(.)$. As we can see from this figure, since $\mu_B(.)$ is the membership function which represents ”n_{keyword} is large”, the larger n_{keyword} takes the value, the closer $\mu_A(n_{\mathrm{keyword}})$ gets to 1. Meanwhile, since $\mu_B^{'}(.)$ is the membership function which represents ”n_{keyword} is small”, the larger n_{keyword} takes the value, the closer $\mu_B^{'}(n_{\mathrm{keyword}})$ gets to 0.

4) Determination of goodness of fit about each condition:

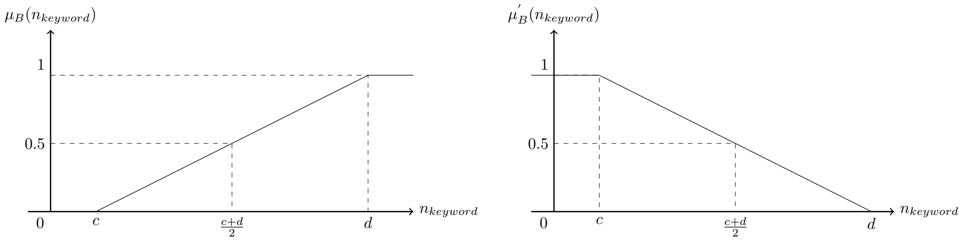
According to m-th proposition, we adapt c_{sub} and n_{keyword} to corresponding membership functions. Let g_{m1} and g_{m2}



(a) $\mu_A(.)$ for condition "c_{sub} is large"

(b) $\mu_A^{'}(.)$ for condition "c_{sub} is small"

Fig. 3: The membership function $\mu_A(.)$ and $\mu_A^{'}(.)$ about c_{sub}



(a) $\mu_B(.)$ for condition "n_{keyword} is large"

(b) $\mu_B^{'}(.)$ for condition "n_{keyword} is small"

Fig. 4: The membership function $\mu_B(.)$ and $\mu_B^{'}(.)$ about n_{keyword}

denote the values of membership functions on m-th proposition. Moreover, G_m that indicates goodness of fit about the proposition m is calculated as

$$G_m = \min(g_{\mathrm{m}_1}, g_{\mathrm{m}_2}). \tag{6}$$

For example, in the proposition (1), we use membership functions $\mu_A(.)$ and $\mu_B(.)$ (i.e. g_{11} = $\mu_A(c_{\mathrm{sub}})$ and g_{12} = $\mu_B(n_{\mathrm{keyword}})$). Since the proposition m consists of two conditions about c_{sub} and n_{keyword}, it is necessary to meet both simultaneously. Thus, the minimum value of g_{m_1} and g_{m_2} is calculated as G_m.

5) *Final score of our feature:*

The score of f acquired from the result of fuzzy inference is calculated as

$$\mathrm{score} = \frac{\max(G_1, G_2) \times 1 + G_3 \times 0 + G_4 \times 0.5}{\max(G_1, G_2) + G_3 + G_4}. \tag{7}$$

Since both conclusion of proposition (1) and (2) are "malicious", we use the larger value of G_1 and G_2. Therefore, $\max(G_1, G_2)$ is goodness of fit about proposition (1) and (2) and it is multiplied by 1. And then, because the conclusion of proposition (3) is "legitimate" and that of proposition (4) is "neutral", G_3 and G_4 are multiplied by 0 and 0.5 respectively. This score takes the value between 0 and 1. The closer this score gets to 1, the more it indicates malicious.

V. Simulation Results

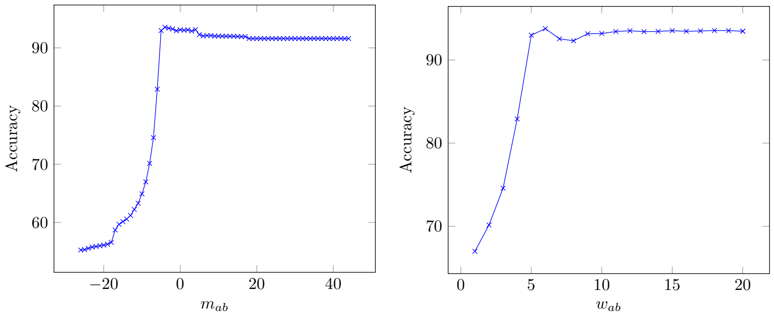
We evaluate our scheme by the computer simulation with real datasets. TABLE II shows our simulation parameters. We get 8045 legitimate PDFs from Contagio [11]. Malicious PDFs are obtained from Contagio and malwere.com [12]. Training data consist of 4586 legitimate PDFs and 5389 malicious PDFs and test data consist of 3459 legitimate PDFs and 4081 malicious PDFs. Moreover, we use AdaBoost [10] as the classifier to evaluate our scheme and the conventional scheme. This is because the conventional scheme uses it. As its implementation, we use scikit-learn [13]. Decision of parameters of membership functions in TABLE II is described in next section. To show effectiveness of our scheme, we evaluate the detection accuracy as

TABLE II: Simulation Parameters

The name of parameters	Value	
Legitimate PDFs	Contagio [11]	
Malicious PDFs	Contagio Malware [12]	
The number of legitimate PDFs	Training data	Test data
	4586	3459
The number of malicious PDFs	Training data	Test data
	5389	4081
Classifier	AdaBoost [10]	
Parameters of membership functions	a = -10, b = 2, c = 22, d = 104	

$$\mathrm{accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{8}$$

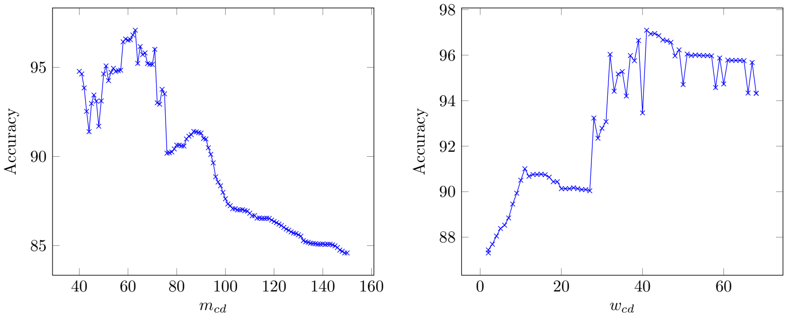
where TP, TN, FP and FN denote the number of True Positive (malicious PDFs are regarded as malicious ones), True Negative (legitimate PDFs are regarded as legitimate ones), False



(a) The accuracy versus $m_{\rm ab}$

(b) The accuracy versus $w_{\rm ab}$

Fig. 5: The accuracy versus $m_{\rm ab}$ and $w_{\rm ab}$



(a) The accuracy versus $m_{\rm cd}$

(b) The accuracy versus $w_{\rm cd}$

Fig. 6: The accuracy versus $m_{\rm cd}$ and $w_{\rm cd}$

Positive (legitimate PDFs are regarded as malicious ones), and False Negative (malicious PDFs are regarded as legitimate ones), respectively.

A. Decision of parameters of membership functions

In order to utilize fuzzy inference, it is necessary to decide parameters of membership functions $\mu_A(.)$, $\mu_A^{'}(.)$, $\mu_B(.)$ and $\mu_B^{'}(.)$. We decide optimal values of a, b, c and d as the following procedure.

As we can see from TABLE I malicious PDFs possess 34 non-duplicate keywords on average and that of a legitimate PDF is 102. Thus, we first set $c = 34$ and $d = 102$. Let $m_{\rm ab}$ and $w_{\rm ab}$ denote the middle point of a and b, and the width of a and b respectively. The membership function in Fig. 3 is parallelly shifted along $c_{\rm sub}$ axis by changing the values of a and b with $w_{\rm ab}$ fixed. Here, $w_{\rm ab} = 10$ is set as initial value. When the detection accuracy is the highest, optimal $m_{\rm ab}$ is determined. Fig. 5 shows the accuracy versus $m_{\rm ab}$ and $w_{\rm ab}$. As illustrated in Fig. 5(a), the optimal $m_{\rm ab}$ is determined as $m_{\rm ab} = -4$. Next, $w_{\rm ab}$ is increased one by one and we decide optimal $w_{\rm ab}$ which brings the highest accuracy rate. As illustrated in Fig. 5(b), the optimal $w_{\rm ab}$ is decided as $w_{\rm ab} = 6$. Therefore, a and b are calculated in the following equation:

$$a = m_{\rm ab} - w_{\rm ab} = -4 - 6 = -10, \tag{9}$$

$$b = m_{\rm ab} + w_{\rm ab} = -4 + 6 = 2. \tag{10}$$

Similarly, we decide parameters of membership functions in Fig. 4. Fig. 6 shows the accuracy versus $m_{\rm cd}$ and $w_{\rm cd}$. As illustrated in Fig. 6(a), when $m_{\rm cd} = 63$, the accuracy rate is the highest. Thus, we determine $m_{\rm cd} = 63$ as optimal value. Moreover, as illustrated in Fig. 6(b), when $w_{\rm cd} = 41$, the accuracy rate is the highest. Thus, c and d are calculated in the following equation:

$$c = m_{\rm cd} - w_{\rm cd} = 63 - 41 = 22, \tag{11}$$

$$d = m_{\rm cd} + w_{\rm cd} = 63 + 41 = 104. \tag{12}$$

In our simulation, we evaluate our scheme by utilizing these parameters mentioned in this Section.

B. Detection Accuracy

We compare the detection accuracy among the conventional and proposed schemes. Conv. is the scheme using conventional scheme's features. Prop.1 is the scheme using conventional scheme's features with proposed scheme's feature. Prop.2 is the scheme using only proposed scheme's feature. Prop.3 is the scheme using conventional scheme's feature with $c_{\rm sub}$ and $n_{\rm keyword}$. Note that Prop.3 is not using Fuzzy inference.

As we can see from TABLE III and TABLE IV, the detection accuracy of Prop.1 is higher than that of Conv. Furthermore, Prop.1 reduces both false positive (legitimate PDFs are regarded as malicious ones) and false negative (malicious PDFs are regarded as legitimate ones) in comparison to the conventional scheme. Since keywords ignored in the conventional scheme are considered by utilizing $c_{\rm sub}$ and $n_{\rm keyword}$, our scheme enables to detect malicious PDFs that the conventional scheme cannot. As a result, our scheme raises the detection accuracy. We find malicious PDFs that possess the keyword /HideWindowUI by inspecting malicious PDFs. These are detected as malicious only in our scheme. This keyword is used in order to manipulate user interfaces in PDF files. Attackers use this in order to perform unnoticed attacks. Therefore, we can say that this keywords is worth to be selected as a feature. However, since this keyword hardly appears in dataset, the conventional scheme ignores this keyword. Therefore, the conventional scheme cannot detect malicious PDFs which possess that. Since our scheme considers such keywords, the malicious PDFs which possess them are regarded as malicious PDFs correctly.

As we can see from TABLE III and TABLE V, the accuracy of the conventional scheme is higher than that of Prop.2. Because the keywords appeared in both legitimate and malicious PDFs are canceled each other, $c_{\rm sub}$ mainly focuses on minor keywords whose number of occurrence is lower than threshold value determined by K-Means clustering algorithm. In comparison to major keywords appeared in both legitimate and malicious PDFs, such keywords are hard to be useful features to detect malicious PDF. Thus, utilizing only minor keywords causes to degrade the detection accuracy. However, since Prop.2 can detect malicious PDFs with high accuracy, we can say that minor keywords can be valuable features to detect malicious PDF.

TABLE III: The result of Conv.

	Predicted		Accuracy (%)
	Legitimate	Malicious	
Legitimate	3445	14	99.23
Malicious	44	4037	

TABLE IV: The result of Prop.1

	Predicted		Accuracy (%)
	Legitimate	Malicious	
Legitimate	3450	9	99.50
Malicious	29	4052	

TABLE V: The result of Prop. 2

	Predicted		Accuracy (%)
	Legitimate	Malicious	
Legitimate	3439	20	98.56
Malicious	87	3994	

TABLE VI: The result of Prop.3

	Predicted		Accuracy (%)
	Legitimate	Malicious	
Legitimate	3450	9	99.43
Malicious	34	4047	

As we can see from TABLE VI and TABLE IV, Prop.1 raises the detection accuracy in comparison to Prop.3. Furthermore, false negative on prop.1 also reduces. This is because Prop.1 can quantitatively represent the maliciousness of a PDF on the situations where the judgement whether malicious or not is unclear. We find the malicious PDF which possesses $c_{\mathrm{sub}} = -1$ and $n_{\mathrm{keyword}} = 41$ by inspecting malicious PDFs which Prop.3 cannot detect. Prop.1 can detect this as malicious. In this case, because the number of keywords appeared in malicious PDF is less than that of legitimate, c_{sub} indicates the legitimate feature. However, since the number of non-duplicate keywords is relatively close to the average value of malicious PDF in TABLE I, n_{keyword} indicates the malicious feature. Thus, because the judgement whether malicious or not is unclear, Prop.3 cannot detect such PDF. On the other hand, because Prop.1 utilizes the feature which fuzzy inference is applied to c_{sub} and n_{keyword}, that can detect it. Therefore, fuzzy inference enables to mitigate the bad effect which influences malicious PDF detection performance.

From these results of our evaluation, since all of schemes can detect malicious PDFs with high accuracy, we can say that utilizing keywords to detect malicious PDFs is considerably effective. Moreover, Prop.1 accomplishes the best detection performance. Thus, using the proposed feature with the conventional scheme's features enables to detect more malicious PDFs.

VI. Conclusion

In this paper, we have proposed the malicious PDF detection scheme using the useful feature based on non-frequent keywords in a file. Our scheme uses two new features about keywords to detect malicious PDFs. This first one is c_{sub} which represents the deference of keywords appeared only in legitimate and malicious dataset. The second one is n_{keyword} which denotes the number of non-duplicate keywords appeared in a file. In order to prevent these features from degrading the detection performance, we apply fuzzy inference to these features and get the score which quantitatively represents maliciousness of a PDF. Our scheme utilizes this with the conventional scheme's features to detect malicious PDFs. By computer simulation with real dataset, we show our scheme can raise the detection accuracy and reduce both false positive and false negative. Furthermore, we will increase the number of dataset and evaluate our scheme in more detail in the future.

Acknowledgment

This work is partly supported by the Grant in Aid for Scientific Research (No.26420369) from Ministry of Education, Sport, Science and Technology, Japan.

References

[1] X. Lu, J. Zhuge, R. Wang, Y. Cao, and Y. Chen, "De-obfuscation and detection of malicious pdf files with high accuracy," in *Proceedings of 46th Hawaii International Conference on System Sciences (HICSS)*, 2013, pp. 4890–4899.

[2] D. Maiorca, I. Corona, and G. Giacinto, "Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious pdf files detection," in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, 2013, pp. 119–130.

[3] C. Smutz and A. Stavrou, "Malicious pdf detection using metadata and structural features," in *Proceedings of the 28th Annual Computer Security Applications Conference*, 2012, pp. 239–248.

[4] D. Maiorca, D. Ariu, I. Corona, and G. Giacinto, "A structural and content-based approach for a precise and robust detection of malicious pdf files," in *Proceedings of the 1st International Conference on Information Systems Security and Privacy (ICISSP)*, 2015, pp. 27–36.

[5] V. Constantin, "Fuzzy logic and neuro-fuzzy applications explained," *Englewood Cliffs, Prentice-Hall*, 1995.

[6] P. Laskov and N. Šrndić, "Static detection of malicious javascript-bearing pdf documents," in *Proceedings of the 27th Annual Computer Security Applications Conference*, 2011, pp. 373–382.

[7] "Pdf reference and adobe extensions to the pdf specification," http://www.adobe.com/devnet/pdf/pdf_reference.html.

[8] D. Liu, H. Wang, and A. Stavrou, "Detecting malicious javascript in pdf through document instrumentation," in *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2014, pp. 100–111.

[9] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967, pp. 281–297.

[10] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Proceedings of European conference on computational learning theory*, 1995, pp. 23–37.

[11] "Contagio," http://contagiodump.blogspot.jp.

[12] "malware," https://malwr.com.

[13] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler *et al.*, "Api design for machine learning software: experiences from the scikit-learn project," *arXiv preprint arXiv:1309.0238*, 2013.