Malware Detection in Portable Document Format (PDF) Files with Byte Frequency Distribution (BFD) and Support Vector Machine (SVM)

Heru Saputra, Deris Stiawan, Hadipurnawan Satria
Department of Computer Engineering, University of Sriwijaya, Indralaya - Ogan Ilir 30662, Indonesia

ARTICLE INFO

Article history:

Received November 06, 2023
Revised December 21, 2023
Published December 27, 2023

Keywords:

Portable document format;
Malware;
Byte frequency distribution;
Sequential forward selection;
Support vector machine

ABSTRACT

Portable Document Format (PDF) files as well as files in several other formats such as (.docx, .hwp and .jpg) are often used to conduct cyber attacks. According to VirusTotal, PDF ranks fourth among document files that are frequently used to spread malware in 2020. Malware detection is challenging partly because of its ability to stay hidden and adapt its own code and thus requiring new smarter methods to detect. Therefore, outdated detection and classification methods become less effective. Nowadays, one of such methods that can be used to detect PDF files infected with malware is a machine learning approach. In this research, the Support Vector Machine (SVM) algorithm was used to detect PDF malware because of its ability to process non-linear data, and in some studies, SVM produces the best accuracy. In the process, the file was converted into byte format and then presented in Byte Frequency Distribution (BFD). To reduce the dimensions of the features, the Sequential Forward Selection (SFS) method was used. After the features are selected, the next stage is SVM to train the model. The performance obtained using the proposed method was quite good, as evidenced by the accuracy obtained in this study, which was 99.11% with an F1 score of 99.65%. The contributions of this research are new approaches to detect PDF malware which is using BFD and SVM algorithm, and using SFS to perform feature selection with the purpose of improving model performance. To this end, this proposed system can be an alternative to detect PDF malware.

Corresponding Author:

Heru Saputra, Department of Computer Engineering, University of Sriwijaya, Indralaya - Ogan Ilir 30662, Indonesia
Email: saputra31.heru@gmail.com

1. INTRODUCTION

In the era of the Industrial Revolution 4.0, the use of digital documents has been considered a common thing to do. This is because many benefits are felt, both in terms of time and cost. The use of digital documents has been widely used in organizations and institutions. One of the digital documents that is often used is Portable Document Format (PDF). PDF contains a combination of text, vector graphics, and raster graphics. It can also contain an audio or video content. Because of its increasingly widespread use among the public. This makes PDF files more often targeted by attackers [1], [2].

Over the years, malware attacks have been increasing rapidly, according to [3], there were billions of files infected with malware in 2021. This makes malware attacks a very serious threat to computer users [4], [5]. According to the 2021 Annual Report on Cyber Security Monitoring issued by BSSN, PDFs as well as several other files such as (.docx, .hwp and .jpg) are very often used to carry out cyber attacks [6], [7]. Furthermore, VirusTotal reported that PDF ranks fourth in document files that are often used to spread *malware*, an increase of 97% compared to the previous year [8].

Malware in PDF files can be embedded in many ways, including using javascript, encoded streams, and embedding an object (image, action code, etc.), which is used to exploit the vulnerability of the PDF reader and then allow the malicious code to run [9], [10], [11]. Some PDF readers and applications are constantly impacted, including CVE-2018-8350 in Microsoft Windows PDF Library and CVE-2017-10994, CVE-2018-14442 in Foxit Reader [12].

PDF Malware detection remains a challenge in cybersecurity. This is because advanced malware is more sophisticated and has the ability to stay hidden or change its code to act smarter [13], [14], [15], [16]. Therefore, outdated detection and classification methods become less effective. As a result, new methods for malware detection are needed, one of which is machine learning [17], [18], [19]. The use of machine learning offers a great deal of simplicity, making it an active domain in the area of cybersecurity [20].

On the other hand, in cyber attacks, one of the methods used to detect infected PDF files is a machine learning approach [21], [22]. Some studies use the Support Vector Machine (SVM) algorithm to detect malware in PDFs [23]. In this research, PDF files will be extracted using PDFiD, a tool that uses the python programming language. However, the sensitivity and precision values are unknown.

Other supported techniques were also used, namely visualization techniques and image processing techniques, to detect malware on PDFs [24]. PDF files are converted into grayscale images for further extraction. Some of the methods used are Random Forest, Decision Tree and K-Nearest Neighbor. The method used is quite complex because PDF files need to be converted first into greyscale images, but the accuracy, sensitivity, and precision values are unknown.

Several studies of PDF malware detection had been done before. Some similar studies used features obtained from the extraction process using the PDFiD tool [23], [25], [26], other tools used by researchers to extract features is PeePDF [27], [28]. Other researchers used visualization techniques and image processing techniques to perform feature extraction [24], [29]. In addition, there were also some researchers who use the byte frequency distribution method to perform feature extraction [30], [31]. Another method used by researchers is using byte streams [32], [33].

The features obtained were then used to train the classification model. This model is formed using several algorithms, some researchers use the SVM algorithm [25], [34], [23]. In addition to using the SVM algorithm, there are also researchers who use machine learning and deep learning methods [35], [32], [36]. Some researchers even combine several methods in their research, such as researchers [37], [38], [39] who combined SVM and CNN methods in their research.

According to [31] regarding file type recognition based on file fragments using statistical methods, researchers compare the accuracy obtained from several algorithms, such as Multilayer perceptron, Support vector machines, and K-Nearest neighbor with the conclusion that the accuracy results obtained from each algorithm were as follows: 95%, 97%, and 98%.

Other research on malware detection in pdf files using the byte stream method was conducted by [32]. In this study, several algorithms were compared, including Decision Tree, Naïve Bayes, Support Vector Machines and Random Forest. The results obtained in this study for the F1 score are as follows: 93.20%, 92.90%, 96.60%, and 96.10%.

In this research, the main objectives are to identify malware detection and determine whether a file is malicious or not. Malware detection is carried out on PDF files, where the data used are private datasets (.pdf, .jpeg and .png) and public datasets (.pdf malware and .pdf benign). The proposed system will use BFD to perform feature extraction, while the SFS method is used for feature selection to reduce the dimensions of the dataset and improve model performance. Researchers will use the SVM algorithm to train the model because of its ability to process non-linear data, and in some studies, SVM produces the best accuracy. The contributions of this research are new approaches to detect PDF malware which is using BFD and SVM algorithm, and the other is using SFS to perform feature selection with the purpose of improving model performance. To this end, this proposed system can be an alternative to detect PDF malware.

The rest of the article is organized as follows: Section 2 presents the methods for the PDF malware detection system. Section 3 presents the performance and experimental evaluation results. Finally, Section 4 provides the conclusion.

2. METHODS

The model developed is able to detect whether the file is: (i) malware PDF files, (ii) non-PDF files, or (iii) benign PDF files. As an overview, the methods used in this research are shown in Fig. 1.

Based on Fig. 1, the PDF malware detection flow began with the dataset preprocessing stage. The next stage is the implementation of SVM for the malware detection process, where the model was created based on the features that had been previously selected. After that, a validation stage is carried out to ensure the accuracy and reliability of the model created. At this stage, an analysis of the validation results is carried out to find out the extent of the model's performance and conclude the research results. In model validation, the author will use the confusion matrix.

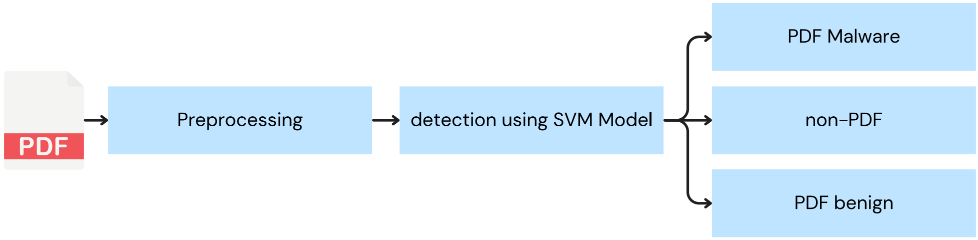


Fig. 1. Overview of PDF Malware detection research

2.1. Dataset and Preprocessing

In this study, datasets were obtained from several sources, namely: private datasets containing pdf files and jpg files, and Contagio datasets containing benign pdf files and pdf malware files. Furthermore, preprocessing was carried out on the data that had been collected using the BFD and SFS methods. The preprocessing flow are described in Fig. 2.

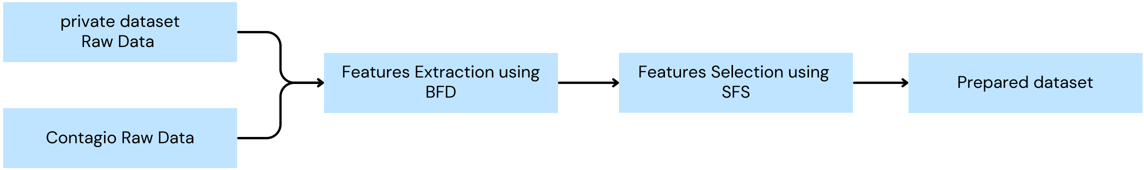
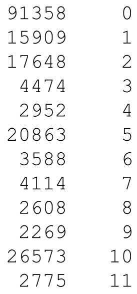


Fig. 2. Dataset preprocessing flow

This preprocessing stage was divided into two, namely the feature extraction process and the feature selection process. The feature extraction process is performed using the Byte Frequency Distribution (BFD) method [40]. In this process, the features were changed to be simpler and easier to interpret by the model. The process carried out was: the file was converted into byte format, and then the frequency of each byte in the file was calculated. As an illustration, the following conversion process was carried out in the Linux operating system using the following command `od -vtu1 -An -w1 filename.pdf | sort -n | uniq -c`. ‘od’ is a command to dump files in octal or another format. It is often used for examining binary files, including executables and documents, while ‘-vtu1 -An -w1’ is the option used to set the output to decimal value. ‘filename.pdf’ is the pdf file name, and ‘sort -n | uniq -c’ used to sort the output numerically and only show unique lines. An example of this command shown in Fig. 3.



```
91358    0
15909    1
17648    2
 4474    3
 2952    4
20863    5
 3588    6
 4114    7
 2608    8
 2269    9
26573   10
 2775   11
```

Fig. 3. The result of converting a .pdf file to byte format (left number of bytes, right byte number)

After obtaining the BFD of a file, feature selection was then carried out using the Sequential Forward Selection (SFS) method with the aim of reducing the dimensions of the dataset and improving model performance [41]. In the process, features were selected and considered iteratively. In each iteration, features continued to be added until the accuracy performance did not increase in the next iteration [42], [43]. This step can be described as follows:

a. The most important features $S1 = fi$ are selected first based on several criteria.
b. Then features are formed with fi and the best feature pair is selected as $S2 =$

$\{fi, fi\}$

c. The third feature set is formed using $S2$, after which the third feature is selected as $S3 = \{fi, fi, fk\}$

d. This process continues to be repeated until the specified number of features are selected.

Fig. 4 shows examples of byte frequency distribution on a .pdf file, the byte value is 0 to 255. While the frequency value has a varying value, up to around 90,000 for the highest score, and around 2,000 for the lowest score. Lastly, the processed data was saved in csv format. The Fig. 5 was an example of a dataset that had been saved in csv format.

Fig. 5 shows examples of first 15 row of dataset, the data contains 24 features and has been labeled, the dataset will be used to train the SVM model.

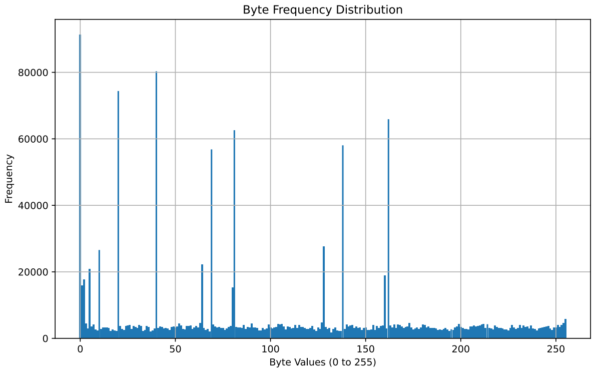
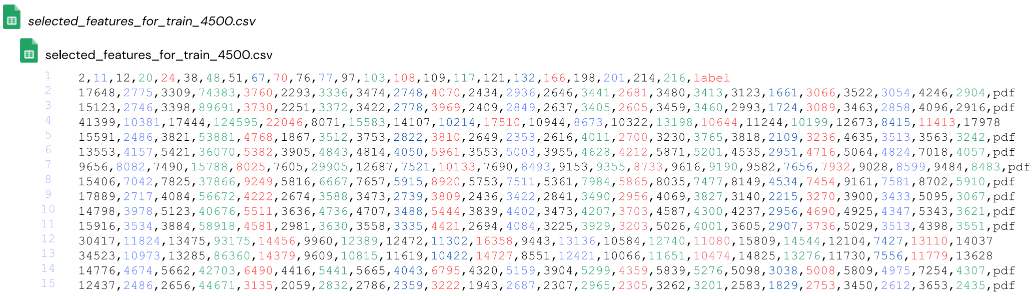


Fig. 4. Example of Byte Frequency Distribution on a .pdf file


selected_features_for_train_4500.csv
selected_features_for_train_4500.csv

```
2,11,12,20,24,38,48,51,67,70,76,77,97,103,108,109,117,121,132,166,198,201,214,216,label
17648,2775,3309,74383,3760,2293,3336,3474,2748,4070,2434,2936,2646,3441,2681,3480,3413,3123,1661,3066,3522,3054,4246,2904,pdf
15123,2746,3398,89691,3730,2251,3372,3422,2778,3969,2409,2849,2637,3405,2605,3459,3460,2993,1724,3089,3463,2858,4096,2916,pdf
41399,10381,17444,124595,22046,8071,15583,14107,10214,17510,10944,8673,10322,13198,10644,11244,10199,12673,8415,11413,17978
15591,2486,3821,53881,4768,1867,3512,3753,2822,3810,2649,2353,2616,4011,2700,3230,3765,3818,2109,3236,4635,3513,3563,3242,pdf
13553,4157,5421,36070,5382,3905,4843,4814,4050,5961,3553,5003,3955,4628,4212,5871,5201,4535,2951,4716,5064,4824,7018,4057,pdf
9656,8082,7490,15788,8025,7605,29905,12687,7521,10133,7690,8493,9153,9355,8733,9616,9190,9582,7656,7932,9028,8599,9484,8483,pdf
15406,7042,7825,37866,9249,5816,6667,7657,5915,8920,5753,7511,5361,7984,5865,8035,7477,8149,4534,7454,9161,7581,8702,5910,pdf
17889,2717,4084,56672,4222,2674,3588,3473,2739,3809,2436,3422,2841,3490,2956,4069,3827,3140,2215,3270,3900,3433,5095,3067,pdf
14798,3978,5123,40676,5511,3636,4736,4707,3488,5444,3839,4402,3473,4207,3703,4587,4300,4237,2956,4690,4925,4347,5343,3621,pdf
15916,3534,3884,58918,4581,2981,3630,3558,3335,4421,2694,4084,3225,3929,3203,5026,4001,3605,2907,3736,5029,3513,4398,3551,pdf
30417,11824,13475,93175,14456,9960,12389,12472,11302,16358,9443,13136,10584,12740,11080,15809,14544,12104,7427,13110,14037
34523,10973,13285,86360,14379,9609,10815,11619,10422,14727,8551,12421,10066,11651,10474,14825,13276,11730,7556,11779,13628
14776,4674,5662,42703,6490,4416,5441,5665,4043,6795,4320,5159,3904,5299,4359,5839,5276,5098,3038,5008,5809,4975,7254,4307,pdf
12437,2486,2656,44671,3135,2059,2832,2786,2359,3222,1943,2687,2307,2965,2305,3262,3201,2583,1829,2753,3450,2612,3653,2435,pdf
```

Fig. 5. Research Dataset in csv format

2.2. Malware detection process using SVM

SVM is a supervised learning method used for classification, regression, and pattern recognition [44], [45]. SVM is based on the concept of breaking data into two classes by determining a hyperplane that separates the two classes by maximizing the distance between the closest data points from the hyperplane, called a vector [46], [47]. SVM has the ability to tackle non-linear problems by projecting the data into a high-dimensional space through kernel tricks. This allows SVMs to solve non-linear problems that cannot be solved by other classification and regression methods. Hyperplane that divides two-dimensional data, as shown in Fig. 6.

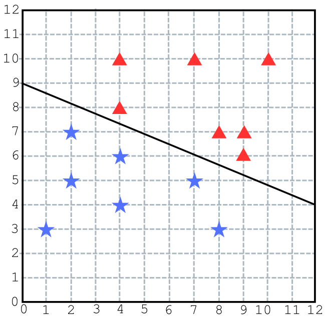


Fig. 6. Hyperplane that divides the data [48]

After the dataset was processed by extracting and selecting features, the next step was to build a model. The model was built using the SVM algorithm. The data were divided into two sets, namely training data (the training set) and test data (the testing set). Training data made up as much as 80% of the total data, while test data made up as much as 20% of the total data. Training data were used to train the SVM model, and test data were used to test the performance of the model.

2.3. Model performance measurement

At this stage, the SVM model that has been built will be measured for performance [49]. The values that will be calculated are: accuracy, sensitivity, precision, and F1 score. To calculate these values, the confusion matrix will be used. The confusion matrix is a table used to describe the performance of the model by comparing the prediction results with the actual value [50]. The confusion matrix has four values, namely True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

Accuracy shows how many predictions are correct compared to the total number of predictions. Accuracy can be calculated by dividing the number of correct predictions by the total number of predictions made. The accuracy calculation equation is as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

Sensitivity measures how well the model can detect positive classes. Sensitivity can be calculated by dividing the number of predicted correct positive results by the total number of actual positive cases. The formula for calculating sensitivity is as follows:

$$Sensitivity = \frac{TP}{TP + FN} \tag{2}$$

Precision shows how well the model can correctly predict the positive class. Precision can be calculated by dividing the number of correct positive predictions by the total number of positive predictions made. The precision calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

The F1 Score is a weighted average comparison of precision and recall. The F1 score provides a balanced average value between sensitivity and precision. F1 score equation to calculate F1 score as follows:

$$F1\ score = \frac{2 * (precision * sensitivity)}{precision + sensitivity} \tag{4}$$

3. RESULTS AND DISCUSSION

At this stage, the results of this research were obtained, and the results were tested by measuring the performance of the model. Implementation was carried out using the Python programming language and using Scikit-Learn library.

3.1. Testing Parameters

In this research, the SVM parameters used were C = 10, gamma = auto, and kernel function rbf. The use of parameters and rbf kernel suggests a moderately flexible decision boundary with an emphasis on the correct detection of training examples. The automatic determination of gamma is chosen to adapt to the characteristics of the input data. By setting these parameters, it was expected that the resulting SVM model could provide accurate and optimal results in accordance with the research objectives.

3.2. Model Testing Results

In testing the model using 4,500 data points consisting of 1,500 PDF files, 1,500 non-PDF files, and 1,500 PDF malware files, the data used was 80% for training and 20% for testing data. The following was a comparison of the testing and training data for each class, as shown in Fig. 7 and Fig. 8.

Fig. 7 shows the percentage of the amount of testing data used, namely 31.67% PDF Malware, 33.34% Non-PDF, and 35% PDF Benign. The total testing data used is 900 data.

Fig. 8 shows the percentage of the amount of testing data used, namely 34% PDF Malware, 33% Non-PDF, and 33% PDF Benign. The total testing data used is 3,600 data.

After several tests, the best results were obtained. The features used for training data are 24, 38, and 43 features. Model testing results are shown in Table 1, Table 2, and Table 3.

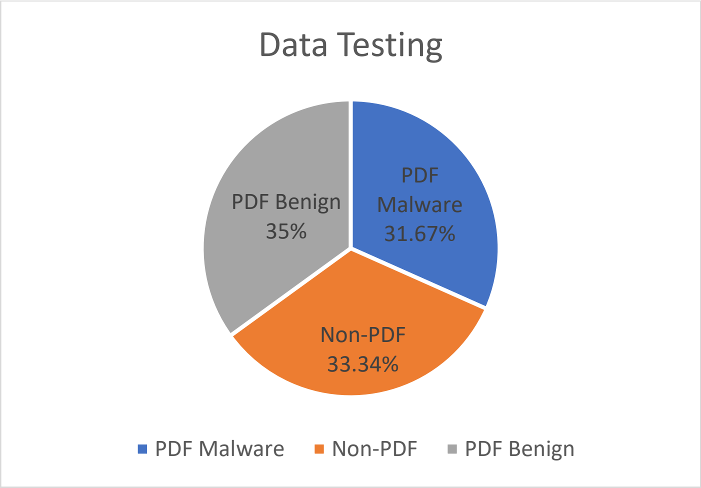


Fig. 7. Comparison of Data Testing

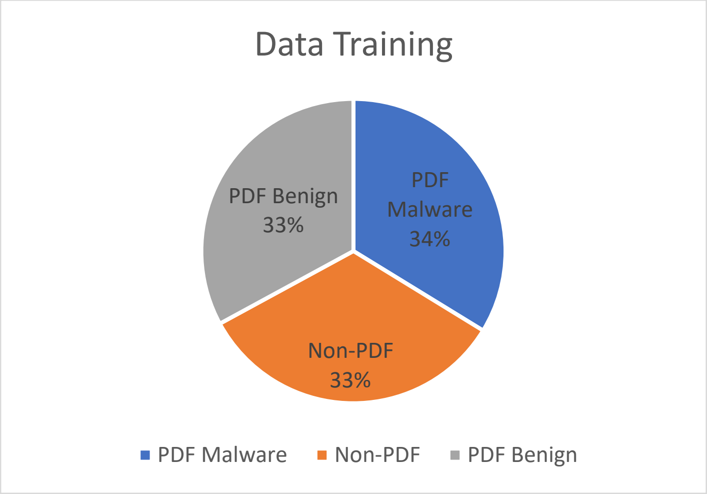


Fig. 8. Comparison of Data Training

3.2.1. First model results

In the first test, the features used to train a total of 24. The results of testing the first model shown in Table 1.

Table 1. First model results

	Precision	Recall	F1 score	Support
PDF Malware	1.0000	0.9930	0.9965	285
Non-PDF	0.9801	0.9867	0.9834	300
PDF Benign	0.9873	0.9873	0.9873	315
accuracy			0.9889	900
macro avg	0.9891	0.9890	0.9891	900
wighted avg	0.9889	0.9889	0.9889	900

The test results of the First Model show that the accuracy of the PDF Malware class scored 98.89%, which means that there are still other classes that are mis-predicted as PDF Malware. The precision score is 100%, the recall score is 99.30%, and the F1 score is 99.65%.

3.2.2. Second model results

In the next test, the features used to train a total of 38 used the same training data samples and test data. The results of testing the first model shown in Table 2.

Table 2. Second model results

	Precision	Recall	F1 score	Support
PDF Malware	1.0000	0.9895	0.9947	285
Non-PDF	0.9736	0.9833	0.9784	300
PDF Benign	0.9841	0.9841	0.9841	315
accuracy			0.9856	900
macro avg	0.9859	0.9856	0.9858	900
wighted avg	0.9856	0.9856	0.9856	900

The test results of the Second Model show that the accuracy of the PDF Malware class scored 98.56%, which means that there are still other classes that are mis-predicted as PDF Malware. The precision score is 100%, the recall score is 98.95%, and the F1 score is 99.47%.

3.2.3. Third model results

In the last test, the features used to train a total of 43 still used the same training data samples and test data. The results of testing the first model shown in Table 3.

Table 3. Third model results

	Precision	Recall	F1 score	Support
PDF Malware	1.0000	0.9930	0.9965	285
Non-PDF	0.9834	0.9900	0.9867	300
PDF Benign	0.9905	0.9905	0.9905	315
accuracy			0.9911	900
macro avg	0.9913	0.9912	0.9912	900
wighted avg	0.9911	0.9911	0.9911	900

The test results of the Third Model show that the accuracy of the PDF Malware class scored 99.11%, which means that there are still other classes that are mis-predicted as PDF Malware. The precision score is 100%, the recall score is 99.30%, and the F1 score is 99.65%.

Three tests were conducted, each using the same training data and test data and using several different features. The following comparison data from the test results is presented in Table 4.

Table 4. Comparison of testing results

Model	Accuracy	Precision	Recall	F1 score
First Model	0.9889	1.0000	0.9930	0.9965
Second Model	0.9856	1.0000	0.9895	0.9947
Third Model	0.9911	1.0000	0.9930	0.9965
[29]	0.973			0.975
[34]	0.9884	0.9880	0.9890	
[19]	0.9965	0.997	0.997	0.997
[26]	0.9989	0.9984	0.9989	0.9986

Table 4 shows that all three models get a score above 80%, both in accuracy and F1 score. The comparison shows that the three models produced are able to detect PDF Malware very well. Table 4 also shows the highest accuracy and F1 score obtained by the third model, this model uses 43 features and the rbf kernel. In the same study comparison, our proposed model get a fairly high score compared to other approaches, although some approaches get a higher score, this is a challenge for the future in order to improve the performance of the model we built.

4. CONCLUSION

In this research, we introduced a new approach to detecting PDF malware. The PDF file was converted into byte format and then presented in BFD. To reduce the dimensions of the features and improve the model performance, the SFS method is used. After the features are selected, the next stage is SVM to train the model. The proposed method achieves good performance, with 99.11% accuracy and 99.65% F1 score. This result is comparable to other approaches in the previous studies. Although the limitation of this research is that this proposed method has not been proven yet to handle evasion attacks. In the future work, we aim to improve model performance by optimizing SVM parameters (the C and gamma parameter) and using different kernels.

REFERENCES

[1] N. Fleury, T. Dubrunquez, and I. Alouani, “PDF-Malware: An Overview on Threats, Detection and Evasion Attacks,” *arXiv preprint arXiv:2107.12873*, 2021, [Online]. Available: http://arxiv.org/abs/2107.12873.

[2] Y. Li, X. Wang, Z. Shi, R. Zhang, J. Xue, and Z. Wang, “Boosting training for PDF malware classifier via active learning,” *Int. J. Intell. Syst.*, vol. 37, no. 4, pp. 2803–2821, 2022, https://doi.org/10.1002/int.22451.

[3] M. Asam *et al.*, “Detection of exceptional malware variants using deep boosted feature spaces and machine learning,” *Appl. Sci.*, vol. 11, no. 21, 2021, https://doi.org/10.3390/app112110464.

[4] B. Vignau, R. Khoury, and S. Halle, “10 Years of IoT Malware: A Feature-Based Taxonomy,” *Proc. - Companion 19th IEEE Int. Conf. Softw. Qual. Reliab. Secur. QRS-C*, pp. 458–465, 2019, https://doi.org/10.1109/QRS-C.2019.00088.

[5] M. S. Akhtar and T. Feng, “Malware Analysis and Detection Using Machine Learning Algorithms,” *Symmetry (Basel).*, vol. 14, no. 11, 2022, https://doi.org/10.3390/sym14112304.

[6] D. Maiorca, B. Biggio, and G. Giacinto, “Towards Adversarial Malware Detection,” *ACM Comput. Surv.*, vol. 52, no. 4, pp. 1–36, 2020, https://doi.org/10.1145/3332184.

[7] BSSN, “Laporan Tahunan Monitoring Keamanan Siber 2021,” 2022. [Online]. Available: https://www.bssn.go.id/laporan-tahunan-monitoring-keamanan-siber-tahun-2021/.

[8] VirusTotal, “Virustotal’s 2021 Malware Trends Report,” 2022. [Online]. Available: https://assets.virustotal.com/reports/2021trends.pdf.

[9] D. Liu, H. Wang, and A. Stavrou, “Detecting malicious javascript in PDF through document instrumentation,” in *Proceedings of the International Conference on Dependable Systems and Networks*, pp. 100–111, 2014, https://doi.org/10.1109/DSN.2014.92.

[10] D. Maiorca and B. Biggio, “Digital Investigation of PDF Files: Unveiling Traces of Embedded Malware,” *IEEE Secur. Priv.*, vol. 17, no. 1, pp. 63–71, 2019, https://doi.org/10.1109/MSEC.2018.2875879.

[11] M. Elingiusti, L. Aniello, L. Querzoni, and R. Baldoni, “PDF-Malware detection: A Survey and taxonomy of current techniques,” *Adv. Inf. Secur.*, vol. 70, pp. 169–191, 2018, https://doi.org/10.1007/978-3-319-73951-9_9.

[12] P. Singh, S. Tapaswi, and S. Gupta, “Malware Detection in PDF and Office Documents: A survey,” *Inf. Secur. J.*, vol. 29, no. 3, pp. 134–153, 2020, https://doi.org/10.1080/19393555.2020.1723747.

[13] S. K. Sahay, A. Sharma, and H. Rathore, "Evolution of Malware and Its Detection Techniques," *Information and Communication Technology for Sustainable Development: Proceedings of ICT4SD,* vol. 933, p. 139, 2019, https://books.google.co.id/books?hl=id&lr=&id=Z0efDwAAQBAJ.

[14] R. Tahir, “A Study on Malware and Malware Detection Techniques,” *Int. J. Educ. Manag. Eng.*, vol. 8, no. 2, pp. 20–30, 2018, https://doi.org/10.5815/ijeme.2018.02.03.

[15] B. Ndibanje, K. H. Kim, Y. J. Kang, H. H. Kim, T. Y. Kim, and H. J. Lee, “Cross-method-based analysis and classification of malicious behavior by API calls extraction,” *Appl. Sci.*, vol. 9, no. 2, 2019, https://doi.org/10.3390/app9020239.

[16] K. Shaukat, S. Luo, and V. Varadharajan, “A novel deep learning-based approach for malware detection,” *Eng. Appl. Artif. Intell.*, vol. 122, p. 106030, 2023, https://doi.org/10.1016/j.engappai.2023.106030.

[17] D. Gibert, C. Mateu, and J. Planes, “The rise of machine learning for detection and classification of malware: Research developments, trends and challenges,” *J. Netw. Comput. Appl.*, vol. 153, p. 102526, 2020, https://doi.org/10.1016/j.jnca.2019.102526.

[18] J. Singh and J. Singh, “A survey on machine learning-based malware detection in executable files,” *J. Syst. Archit.*, vol. 112, p. 101861, 2021, https://doi.org/10.1016/j.sysarc.2020.101861.

[19] S. Y. Yerima, A. Bashar, and G. Latif, “Malicious PDF detection Based on Machine Learning with Enhanced Feature Set,” in *Proceedings 14th IEEE International Conference on Computational Intelligence and Communication Networks, CICN*, pp. 486–491, 2022, https://doi.org/10.1109/CICN56167.2022.10008374.

[20] F. A. Aboaoja, A. Zainal, F. A. Ghaleb, B. A. S. Al-rimy, T. A. E. Eisa, and A. A. H. Elnour, “Malware Detection Issues, Challenges, and Future Directions: A Survey,” *Appl. Sci.*, vol. 12, no. 17, 2022, https://doi.org/10.3390/app12178482.

[21] M. J. Hossain Faruk *et al.*, “Malware Detection and Prevention using Artificial Intelligence Techniques,” *Proc. - IEEE Int. Conf. Big Data, Big Data*, pp. 5369–5377, 2021, https://doi.org/10.1109/BigData52589.2021.9671434.

[22] S. S. Alshamrani, “Design and Analysis of Machine Learning Based Technique for Malware Identification and Classification of Portable Document Format Files,” *Secur. Commun. Networks*, 2022, https://doi.org/10.1155/2022/7611741.

[23] B. Cuan, A. Damien, C. Delaplace, and M. Valois, “Malware detection in PDF files using machine learning,” *ICETE 2018 - Proc. 15th Int. Jt. Conf. E-bus. Telecommun.*, vol. 2, pp. 412–419, 2018, https://doi.org/10.5220/0006884704120419.

[24] A. Corum, D. Jenkins, and J. Zheng, “Robust PDF Malware Detection with Image Visualization and Processing Techniques,” *Proc. 2nd Int. Conf. Data Intell. Secur. ICDIS 2019*, pp. 108–114, 2019, https://doi.org/10.1109/ICDIS.2019.00024.

[25] A. Charim, S. Basuki, and D. R. Akbi, “Detect Malware in Portable Document Format Files (PDF) Using Support Vector Machine and Random Decision Forest,” *J. Online Inform.*, vol. 3, no. 2, p. 99, 2019, https://doi.org/10.15575/join.v3i2.196.

[26] M. Issakhani, P. Victor, A. Tekeoglu, and A. Lashkari, “PDF Malware Detection based on Stacking Learning,” in *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 562–

570, 2022, https://doi.org/10.5220/0010908400003120.

[27] S. Selvaganapathy and S. Sadasivam, “Malware Attacks on Electronic Health Records,” in *Congress on Intelligent Systems*, pp. 589–599, 2021, https://doi.org/10.1007/978-981-33-6981-8_47.

[28] P. P. Chandran, N. Hema Rajini, and M. Jeyakarthic, “Intelligent Optimal Gated Recurrent Unit based Malicious PDF Detection and Classification Model,” *Proc. - Int. Conf. Appl. Artif. Intell. Comput. ICAAIC,* pp. 1273–1279, 2022, https://doi.org/10.1109/ICAAIC53929.2022.9793116.

[29] C.-Y. Liu, M.-Y. Chiu, Q.-X. Huang, and H.-M. Sun, “PDF Malware Detection Using Visualization and Machine Learning,” in *Data and Applications Security and Privacy XXXV*, pp. 209–220, 2021, https://doi.org/10.1007/978-3-030-81242-3_12.

[30] G. Y. Kim, J. Y. Paik, Y. Kim, and E. S. Cho, “Byte Frequency Based Indicators for Crypto-Ransomware Detection from Empirical Analysis,” *J. Comput. Sci. Technol.*, vol. 37, no. 2, pp. 423–442, 2022, https://doi.org/10.1007/s11390-021-0263-x.

[31] M. Masoumi, A. Keshavarz, and R. Fotohi, “File fragment recognition based on content and statistical features,” *Multimed. Tools Appl.*, vol. 80, no. 12, pp. 18859–18874, 2021, https://doi.org/10.1007/s11042-021-10681-x.

[32] Y. S. Jeong, J. Woo, and A. R. Kang, “Malware Detection on Byte Streams of PDF Files Using Convolutional Neural Networks,” *Secur. Commun. Networks*, 2019, https://doi.org/10.1155/2019/8485365.

[33] Y. S. Jeong, S. M. Lee, J. H. Kim, J. Woo, and A. R. Kang, “Malware Detection Using Byte Streams of Different File Formats,” *IEEE Access*, vol. 10, pp. 51041–51047, 2022, https://doi.org/10.1109/ACCESS.2022.3171775.

[34] Q. Al-Haija, A. Odeh, and H. Qattous, “PDF Malware Detection Based on Optimizable Decision Trees,” *Electron*, pp. 562–570, 2022, https://doi.org/10.5220/0010908400003120.

[35] S. Dey, A. Kumar, M. Sawarkar, P. K. Singh, and S. Nandi, "EvadePDF: Towards evading machine learning based PDF malware classifiers," *In Security and Privacy: Second ISEA International Conference*, pp. 140-150, 2019, https://doi.org/10.1007/978-981-13-7561-3_11.

[36] J. Zhang, “MLPdf: An Effective Machine Learning Based Approach for PDF Malware Detection,” *arXiv preprint arXiv:1808.06991*, 2018, [Online]. Available: http://arxiv.org/abs/1808.06991.

[37] K. He, Y. Zhu, Y. He, L. Liu, B. Lu, and W. Lin, “Detection of malicious PDF files using a two-stage machine learning algorithm,” *Chinese J. Electron.*, vol. 29, no. 6, pp. 1165–1177, 2020, https://doi.org/10.1049/cje.2020.10.002.

[38] S. S. Lad and A. C. Adamuthe, “Malware classification with improved convolutional neural network model,” *Int. J. Comput. Netw. Inf. Secur.*, vol. 12, no. 6, pp. 30–43, 2020, https://doi.org/10.5815/ijcnis.2020.06.03.

[39] A. Bensaoud, N. Abudawaood, and J. Kalita, “Classifying Malware Images with Convolutional Neural Network Models,” *Int. J. Netw. Secur.*, vol. 22, no. 6, pp. 1022-1031, 2020, https://doi.org/10.6633/IJNS.202011_22(6).17.

[40] S. S. Mousavi, “Detecting Disk Sectors Data Types Using Hidden Markov Model,” in *Proceedings of 17th International ISC Conference on Information Security and Cryptology, ISCISC*, pp. 60–64, 2020, https://doi.org/10.1109/ISCISC51277.2020.9261906.

[41] M. Pahar, M. Klopper, R. Warren, and T. Niesler, “COVID-19 cough classification using machine learning and global smartphone recordings,” *Comput. Biol. Med.*, vol. 135, p. 104572, 2021, https://doi.org/10.1016/j.compbiomed.2021.104572.

[42] B. Venkatesh and J. Anuradha, “A review of Feature Selection and its methods,” *Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, 2019, https://doi.org/10.2478/CAIT-2019-0001.

[43] H. Polat and O. Polat, “Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models,” *Mdpi*, vol. 2, no. 3, p. 1035, 2020, https://doi.org/10.3390/su12031035.

[44] R. El-Sayed, A. El-Ghamry, T. Gaber, and A. E. Hassanien, “Zero-Day Malware Classification Using Deep Features with Support Vector Machines,” in *Proceedings - IEEE 10th International Conference on Intelligent Computing and Information Systems, ICICIS*, pp. 311–317, 2021, https://doi.org/10.1109/ICICIS52592.2021.9694256.

[45] Y. F. Lu, C. F. Kuo, H. Y. Chen, C. W. Chen, and S. C. Chou, “A SVM-Based Malware Detection Mechanism for Android Devices,” *2018 Int. Conf. Syst. Sci. Eng. ICSSE,* pp. 1–6, 2018, https://doi.org/10.1109/ICSSE.2018.8520241.

[46] L. Ghouti and M. Imam, “Malware classification using compact image features and multiclass support vector machines,” *IET Inf. Secur.*, vol. 14, no. 4, pp. 419–429, 2020, https://doi.org/10.1049/iet-ifs.2019.0189.

[47] A. B. Yilmaz, Y. S. Taspinar, and M. Koklu, “Classification of Malicious Android Applications Using Naive Bayes and Support Vector Machine Algorithms,” *Int. J. Intell. Syst. Appl. Eng.*, vol. 10, no. 2, pp. 269–274, 2022, https://doi.org/10.1039/b000000x.

[48] A. Kowalczyk. *Support Vector Machines Succinctly*. Syncfusion Inc, 2017, https://www.syncfusion.com/succinctly-free-ebooks/support-vector-machines-succinctly.

[49] S. A. Hicks *et al.*, “On evaluation metrics for medical applications of artificial intelligence,” *Sci. Rep.*, vol. 12, no. 1, pp. 1–9, 2022, https://doi.org/10.1038/s41598-022-09954-8.

[50] A. Tharwat, “Classification assessment methods,” *Appl. Comput. Informatics*, vol. 17, no. 1, pp. 168–192, 2018, https://doi.org/10.1016/j.aci.2018.08.003.

BIOGRAPHY OF AUTHORS

Heru Saputra, currently a Master's student in Universitas Sriwijaya. He received her undergraduate degree in the same university, majoring in Informatics. He areas of interest include Crypthography, Machine Learning, and Cyber Security. He can be contacted at email: saputra31.heru@gmail.com.

Deris Stiawan, received the PhD degree in Computer Engineering from Universiti Teknologi Malaysia, Malaysia. He is currently a Professor at Department of Computer Engineering, Faculty of Computer Science, Universitas Sriwijaya. His research interests include computer network, Intrusion Detection/ Prevention System, and heterogeneous network. He can be contacted at email: deris@unsri.ac.id.

Hadipurnawan Satria, received the PhD degree in Computer Science from Sun Moon University, South Korea. He is currently a Lecturer at Department of Computer Engineering, Faculty of Computer Science, Universitas Sriwijaya. His research interests include Platform-based Development, Embedded System, and Software Engineering. He can be contacted at email: hadi@ilkom.unsri.ac.id.