File fragment recognition based on content and statistical features

Marzieh Masoumi[1]. Ahmad Keshavarz[2]

[1]Deparment of Computer Engineering, Tehran Science and Research Branch, Islamic Azad University, Bushehr, Iran.

[2]Electrical Engineering Department, School of Engineering, Persian Gulf University, Bushehr, Iran.

Email: marzyeh_maasoumi@yahoo.com, a.keshavarz@gpu.ac.ir

Abstract: Nowadays, speed up development and use of digital devices such as smartphones have put people at risk of internet crimes. The evidence of present crimes in a computer file can be easily unreachable by changing the prefix of a file or other algorithms. In more complex cases, either file divided into different parts or the parts of a file that has information about the file type are deleted, where the file fragment recognition issue is discussed. The known files are divided into different fragments, and different classification algorithms to solve the problems of file fragment recognition. A confusion matrix measures the accuracy of type recognition. In the present study, first, the file is divided into different fragments. Then, the file fragment features, which are obtained from Binary Frequency Distribution (BFD), are reduced by 2 feature reduction algorithms; Sequential Forward Selection algorithm (SFS) as well as Sequential Floating Forward Selection algorithm (SFFS) to delete sparse features that result in increased accuracy and speed. Finally, the reduced features are given to 3 classifier algorithms, Multilayer Perceptron (MLP), Support Vector Machines (SVM), and K-Nearest Neighbor (KNN) for classification and comparison of the results. In this paper, we proposed the algorithm of file type recognition that can recognize 6 types of useful files (pdf, txt, jpg, doc, html, exe), which may distinguish a type of file fragments with higher accuracy than the similar works done

Keywords: Classification Algorithm, Feature Reduction, File Fragments, File Fragment Recognition, SFS, SFFS.

1. Introduction

Computers deal with a large number of file with different formats, which are transmitted among networks. The format of a file is an initial design of it that tells the processor devices how to organize the file information and describe their decoding algorithm in digital storage devices. The security of computers and networks reduces without the correct detection of the file type. Detecting the file type is a significant step in adequate proceed of operating systems, firewalls, intrusion detection systems, and anti-viruses.

The content-based algorithm includes investigating the file content and using static techniques. The contents of the file are a chain of bytes, and each byte has 256 unique characters (0-255). Therefore, the calculation of the byte pattern rate, referred to as the byte distribution rate provides a recognizable pattern for different file types.

McDaniel and Heidari [1] were the first to develop an algorithm for recognizing the file types based on content. Their proposed algorithms are used to generate a "fingerprint" of each file, which are detected compared with the known types, and file types. The accuracy varies between 23% and 96% depending on the algorithm used.

Li et al. [2] made slight changes to the McDaniel's model, which increased its accuracy. They provided a set of central models and used the categorization to find the minimum number of centers set with good performance while using more data patterns. This research has the accuracy of 82% (single central) and 89.5% (multi-center) with 93.5% of more sample files.

Karresand and Shahmehri [3] provided an algorithm for file fragments, which used the BFD and the standard deviation concept for file type modeling. Karresand and Shahmehri proposed the Oscar methodology for detecting the file fragments. They generated single-center printing files but used a quadratic distance metric and a norm-1 as the metric distance to compare the center with the byte frequency distribution of the file. Although Oscar recognized any file type, they reported their algorithms for jpg files using the specified pair bytes of the optimized file and the detection rate of 99.2%.

Veenman [4] extracted three features from the file's content. These features include:

1. Frequency byte distribution
2. The entropy obtained by frequency byte distribution of files
3. The complexity of the algorithm or the Kolmogorov that uses the sequence of the substring

Fisher’s linear discriminant analysis has been applied to these features to recognize the file types.

Calhoun and Coles [5] used a static algorithm and the linear FISHER one for a dataset containing 100 fragments of 2 different file types with an accuracy of 60.3% -86%

(depending on the tested bytes chain). They have developed the Veenman works by the constructed classification models and presented the linear discriminant to recognize the file types. Further, they have examined machine learning algorithms to solve the data classification problem and achieved a reasonable accuracy.

Sportielo and Zanero [6] have considered a set of SVM classifications for each file type. The results of several experiments show that the features based on the byte frequency distribution have the best performance for most of the examined file types, where the SVM is very effective in distinguishing file types from the data blocks.

Gopal et al. [7] introduced the File Type Recognition (FTI) as a significant issue in digital rules and provided a systematic review of the problem, algorithmic solutions, and evaluation methodologies. They analyzed the power of various algorithms in examining the files and damaged fragments. They also proposed two criteria for replacement in performance measurement as follows:

1. Considering the file name extension as the correct tags (labels)
2. Considering the prediction by knowledge-based algorithms in healthy files as the correct tags (labels)

The conclusion was that the SVM and KNN are better than COTS (Commercial off-the-shelf) in files that the extensions for sound files are available. Also, some COTS algorithms can detect the corrupted files by no means.

Moody and Erbacher [8] used the static analysis to recognize the file type (SADI), which includes the mean, standard deviation, average distance, standard deviation distance, and calculation of the bytes values. They used the fragments of 200 files from a dataset of 8 known files, which had a 74.2% result.

Dunham et al. [9] applied the neural networks for categorizing ten file types from a dataset, including 760 archived files with an accuracy of 91.5%.

Like et al. [10] adapted the BFD model with the Manhattan distance for comparison to determine whether the calculated files are executable or not.

Cao et al. [11] used the Gram frequency distribution and the vector space model with a 40.34% result.

Ahmad et al. [12] presented two algorithms. First, they applied the cosine distance as a metric of similarity when comparing the file contents. Secondly, they divided the recognition process into two steps by Dividing and Conquering algorithms. In the first step, the similar files with the same byte frequency patterns are classified in different clusters. In the next phase, the classification, including various file types, is given to the neural networks to improve the categorization. They used 2000 different file types with the accuracy of 90.19%.

Ahmad et al. [13] also proposed two new algorithms to reduce the classification time. First, they used the Feature Selection technique and KNN classifier. The second algorithm was the sample content technique in which they used a small portion of the file to achieve the byte frequency distribution.

As described in this section, many works have been done in this approach, but then again, unfortunately, they did not specify their datasets. Moreover, they used both different types of files and datasets, which caused impossible conditions to compare them correctly with each other.

In 2015, in an experiment, Nasser Alamri [14] compared six different file types (pdf, txt, jpg, doc, html, and exe) with 5 algorithms presented on the specific database, and then provided the way of comparison in future studies. We also chose Nasser Alamri's article to compare the suggested algorithms. Thus, we applied the same database and file types with Alamri [14] that provides a reasonable comparison for the present study. The purpose of this research was to recognize the file fragment types with higher accuracy than the similar research works due to the widespread use of this issue as well as its sensitivity to the correct recognizing file type. In the following, the dataset, the methodology of the proposed algorithm, and the obtained results were described. Finally, the results of this study were compared with the results of Naser Alamri study in 2015.

2. Methodology

The train and test sets provided by dividing a file into small fragments. Hence, we fragment complete files, but at first, we cut the header and prefix of files, which may contain information about files type. Then, we divided the rest of each file into 2 fragments of 500 and 1000 bytes, to show the effect of fragment size on the accuracy of the presented methods in the study.

As illustrated in figure 1, SFS and SFFS algorithms were used to reduce the fragment size of the studied file and select the dynamic features. The KNN, SVM, and MLP algorithms were employed as file type detection algorithms. The LIBSVM Package was employed for SVM classification and, MATLAB Toolbox AutoEncoder was utilized for the neural network.

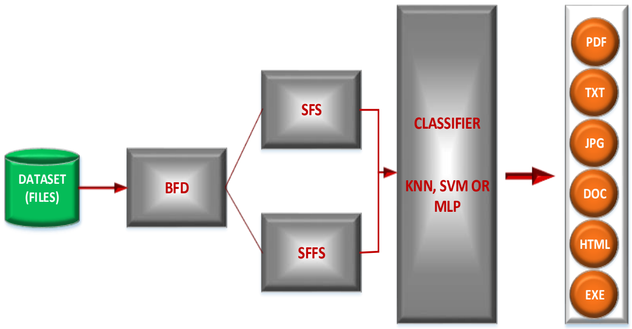

Fig.1. The Methodology implemented in this work

2.1 BFD Extraction and BFD Normalization

The byte frequency distribution (BFD) was used as the feature extraction algorithm. After obtaining the array bytes values rate, each member of the array was distributed by the byte frequency rate. Accordingly, the array was normalized to values between 0 and 1. Figure 2 displays the BFD diagram for the 500-byte fragments and Figure 3 further shows the BFD diagram for 1000-byte fragments.

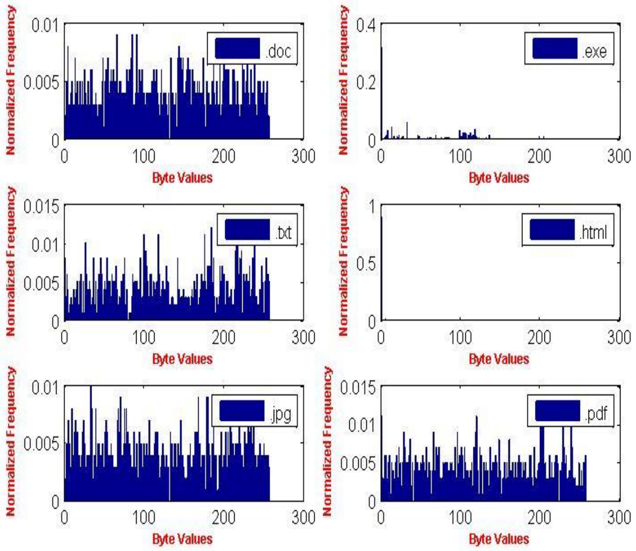

Fig.2. The BFD graph for the 500-fragment

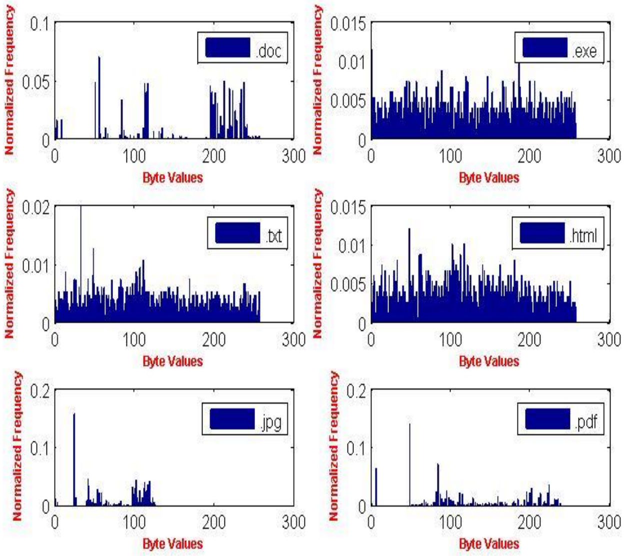

Fig.3. The BFD graph for the 1000-fragment

2.2 Feature Reduction

Different feature selection algorithms strain to find the best subset among the n^2 candidate subsets. These algorithms increase the accuracy and speed by eliminating the outliers. We adopted SFS and SFFS algorithms in the present study as feature reduction algorithms. We tried several parameters for k, and finally, the KNN algorithm with K = 5 was considered as the criterion for feature selection. With the algorithms mentioned, 256 features, which obtain from BFD, were reduced to 24 and 39 features. We performed the feature selection process on all 500 and 1000 byte fragments. The corresponding results are given in Table 2.

2.2.1 Preliminaries

In advance of describing the corresponding algorithms formally, the following definitions have to be introduced. Let $X_k = \{X_i : 1 \leq X_i \leq K, X_i \in Y\}$ be the set of k features from the set $Y = \{y_i : 1 \leq i \leq D\ \}$ of D available features.

The value $J\ (y_i)$ of the feature selection criterion function if only the i_{th}feature $y_i = (i = 1,2,)$ used will be called the individual significance $S_0(y_j)$ of the feature.

The significance $S_{k-1}(x_j)$of the feature $X_j,\ j = 1,2,\dots,k$ in the set X_k is defined by:

$$S_{k-1}(X_j) = J(X_k) - J(X_k - X_j)$$

The significance $S_{k+1}(f_j)$ of the feature $f_j$from the set $Y - X_k$

$$Y - X_k = \{f_i : i = 1,2,\dots,D-k, f_i \in Y, f_i \neq X_t\ for\ all\ x_t \in X_k\}$$

So, $X_k$is defined by

$$S_{k+1}(X_j) = J(X_k + f_j) - J(X_k)$$

For K = 1 the term feature significance in the set coincides with the term of individual significance.

We shall say that the feature x_j (b) from the set $X_k$is (a) the most significant (best) feature in the set X_k if

$$S_{k-1}(X_j) = \max_{1\le i\le k} S_{k-1}(X_i) \Rightarrow J(X_k - x_j)$$
$$= \min_{1\le i\le k} J(X_k - x_i)$$

(b) The least significant (worst) feature in the set X_k if

$$S_{k-1}(X_j) = \min_{1\le i\le k} S_{k-1}(X_i) \Rightarrow J(X_k - x_j)$$
$$= \max_{1\le i\le k} J(X_k - x_i)$$

We shall say that the feature f_j from the set $Y - X_k$ is (a) the most significant (best) feature with the set X_k if

$$S_{k+1}(f_j) = \max_{1\le i\le D-k} S_{k+1}(f_i) \Rightarrow J(X_k + f_j)$$
$$= \max_{1\le i\le D-k} J(X_k + f_i)$$

(b) The least significant (worst) feature concerning the set X_k if

$$S_{k+1}(f_j) = \min_{1\le i\le D-k} S_{k+1}(f_i) \Rightarrow J(X_k + f_j)$$
$$= \min_{1\le i\le D-k} J(X_k + f_i)$$

2.2.2 Sequential Forward Selection (SFS) Algorithm

In the "sequential feature selection" (SFS) algorithm, the process starts with an empty set. Then, in each repetition, a feature is added to the answer set by employing the evaluation function used. This is repeated until the selection of the required features [15]. Using SFS, we achieved 24 features and 36 for 500-byte and 1000-byte fragments, respectively.

1. start with an empty set $y_0 = \{\emptyset\}$
2. Choose the next best features

   $x^+ = arg_{x\notin y_k}\, max\, j(y_k + x)$

   Update set

   $x^+ = arg_{x\notin y_k}\, max\, j(y_k + x)$
3. Return to step 2

2.2.3 Sequential Floating Forward Selection (SFFS) Algorithm

First, the sequential floating forward selection (SFFS) algorithm begins with an empty set of features. For each step, the best feature that satisfies the criterion function is placed in the current set. That is, one stage of the sequential forward selection is performed. The SFFS progresses with dynamic increasing or decreasing of the feature numbers to achieve the optimal number of them [16]. Using SFFS, we obtained 36 and 39 features for the 500-byte and 1000-byte fragments, respectively.

1. start with an empty set $y_0 = \{\emptyset\}$
2. Choose the next best features

   Update set $k = k + 1$ و $y_{k+1} = y_{k+1} + x^+$
3. Choose the worst features $x^- = arg_{x\notin y_k}\, max\, j(y_k - x)$
4. If $j(y_k - x) > y_k$

   $k = k + 1$ و $y_{k+1} = y_k - x^-$

   Return to step 3

   Else

   Return to step 2

2.3 Classification

At the stage of categorizing the type of file fragments, the acquired features are used as inputs in three algorithms, KNN, SVM, and MLP as described below.

The KNN algorithm is a simple supervised algorithm that stores all available cases in different categories based on a similarity measure and classifies new cases [17]. The k parameter displays the number of closest neighbors in the feature space. We used a KNN algorithm with k = 4, 6, 8, and 10; the results are illustrated in table 3 for 1000-byte and table 5 for 500-byte fragments.

The SVM algorithm is a supervised algorithm that performs classification by finding the hyperplane, which maximizes the margin between the two classes [17]. In this study of file fragment recognition, we use SVM algorithm as the second classification approaches with Radial Basis Function (RBF) kernel as well as a different c parameter, c = 0.1, 0.2, 0.3, as shown in table 3 for 1000-byte and table 5 for 500-byte fragments.

The MLP is the third classification algorithm used in the study. It is a type of feedforward neural network, which may differentiate data that is not linearly separable [17]. We use MLP with 1 hidden layer and sigmoid activation function as shown in Figure 4 and the result in Table 3 for 1000-byte and table 5 for 500-byte fragments.

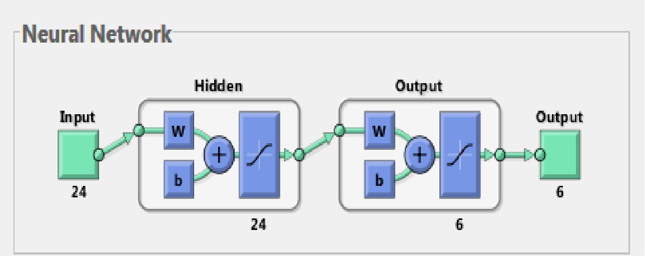

Fig.4.The MLP used model in this study

We tried several parameters in each algorithm to obtain the best result. The corresponding results are given in Table 4 for 1000-byte and Table 6 for 500-byte fragments.

3. Dataset

The standardized Govdocs1 dataset, containing 1,000 lists of 1,000 content files, was used in this research. From 3 random folders of this database, we extracted 100 files from each sample of txt, jpg, htm, and pdf (totally 600 files) with a minimum size of 4Kb. The exe files were obtained from Windows system files by considering their minimum size. The data applied in the program are standard data that are used extensively in similar studies; these are available at the following address: http://digitalcorpora.org/corpora/govdocs.

In the present research, we focused on the file types included in the dataset section; the statistical descriptions are given in Table 1.

Table 1. **files types**

average size	maximum size	minimum size	number	type
345,796	9,023,488	12,800	100	doc
187,498	6,440,448	4,724	100	exe
391,526	1,063,025	4,061	100	txt
76,370	16,497,395	4,008	100	htm
162,012	7,778,639	4,023	100	jpg
608,778	10,891,418	4,710	100	Pdf

4. Implementation Results

In this section, the results obtained from the implementation are analyzed, and finally, the result of the proposed algorithm is compared with other available algorithms. We presented the results of the implementation of the proposed solution in two parts of 1000-byte and 500-byte fragments. For both 1000-byte and 500-byte fragments, we reduced the features obtained from BFD components via SFS and SFFS algorithms. At that point, we gave the reduced set of features to the SVM, KNN, and MLP classifier algorithms. The accuracies of the classifiers are given in the tables. The results presented in the tables are the outcomes of 10 repetitions of the algorithm with various parameters. The best result of each classifier algorithm with a different combination of the training rate and the corresponding parameters of the feature reduction algorithm for 1000-item fragments are shown in Table 3.

Table 2. **Feature reduced results**

The Number of features selected For 1000 fragments	The Number of features selected For 500 fragments	algorithm
36	24	**SFS**
39	35	**SFFS**

Table 3. **Results of 1000 fragments**

parameter	classifier	algorithm	Number of features	Train/test	accuracy
	MLP	SFFS	39	90/10	95%
K=4	KNN	SFS	36	90/10	96%
K=6	KNN	SFS	36	90/10	97%
K=8	KNN	SFFS	39	90/10	97%
K=10	KNN	SFS	36	90/10	97%
C=0.1	SVM	SFFS	39	90/10	97%
C=0.2	SVM	SFFS	39	90/10	98%
C=0.3	SVM	SFFS	39	90/10	98%

The best results obtained in 1000-byte fragments with the best possible combinations are given in Table 4 below.

Table 4. **The best results of 1000 fragments**

classifier	algorithm	Number of features	Train/test	accuracy
MLP	SFFS	39	90/10	95%
KNN	SFS و SFFS	36 or 39	90/10	97%
SVM	SFFS	39	90/10	98%

According to Table 4, the MLP algorithm with 96% accuracy, the KNN algorithm with an accuracy of 97%, and the SVM algorithm with an accuracy of 98% completed their process in the 1000-byte fragments. Accordingly, the SVM algorithm is considered the best algorithm for recognizing the 1000-byte files with an accuracy of 98%.

The best result of each classifier algorithm with a different combination of the training rate and the corresponding parameters of the feature reduction algorithm for 500-item fragments are shown in Table 5.

Table 5. Results of 500 fragments

parameter	classifier	algorithm	Number of features	Train /test	accuracy
	MLP	SFFS	35	90/10	96%
K=4	KNN	SFFS	35	90/10	98%
K=6	KNN	SFFS	35	90/10	98%
K=8	KNN	SFFS	35	90/10	98%
K=10	KNN	SFFS	35	90/10	98%
C=0.1	SVM	SFFS	35	90/10	98%
C=0,2	SVM	SFFS	35	90/10	97%
C=0,3	SVM	SFFS	35	90/10	98%
C=0.4	SVM	SFFS	35	90/10	98%

The best results obtained in 500-byte fragments with the best possible combinations are given in Table 6.

Table 6. The best results of 500 fragments

classifier	algorithm	Number of features	Train /test	accuracy
MLP	SFFS	35	90/10	96%
KNN	SFS & SFFS	36 or 39	90/10	98%
SVM	SFFS	39	90/10	98%

According to Table 6, the MLP algorithm with 95% accuracy, the KNN algorithm with an accuracy of 98%, and the SVM algorithm with an accuracy of 98% completed their process in the 500-byte fragments. Accordingly, the KNN and SVM algorithms are considered the best algorithms for recognizing the 500-byte files with an accuracy of 98%.

5. Analysis of the Research Results

The best results of the research by comparing two SFS and SFFS algorithms, as well as both 500-byte and 1000-byte fragments, are presented in Table 7.

Table 7. Results to be compared

classifier	Number of features	Fragment size	Train /test	accuracy
MLP-s	35	500 Byte	90/10	96%
K-NN-s	35	500 Byte	90/10	98%
SVM-s	35	500 Byte	90/10	98%

Referring to Table 7, the MLP algorithm provides its best result on the 500-byte fragments with SFFS feature reduction algorithm by selecting 35 features. The best result recorded for the MLP algorithm in this study is 96%. The

KNN algorithm also provides its best result on the 500-byte fragments with the SFFS feature reduction algorithm by selecting 35 features. The best result recorded for the KNN algorithm in the current study is 98%. The SVM algorithm also provides its best result on the 500-byte fragments with the SFFS feature reduction algorithm by selecting 35 features. The best result recorded for the SVM algorithm in this research is 98%. We called these proposed algorithms SVM-s, KNN-s, and MLP-s, respectively.

As specified by the results, by increased length of the fragments from 500 to 1000 bytes, the examined algorithms provide either weaker or similar results with a minimal alteration, which can be due to a small difference in the number of features obtained from SFS and SFFS reductions algorithms for 1000-bytes fragments compared to 500-byte fragments.

As illustrated in Table 7, the SVM and KNN algorithms with similar accuracy of 98% are at the highest place, and the MLP algorithm with an accuracy of 96% occurs in a lower place. This means feature reduction by SFFS algorithm will provide better results than the SFS algorithm for 1000-byte and 500-byte fragments. Moreover, the SVM and KNN algorithms have a better performance than the MLP algorithm.

6. Comparison of the Proposed Algorithm with other Algorithms

The study in the field of recognizing the file type includes a large number of file types as well as different databases. This leads to complexity in the comparison and conclusion of the research. In 2015, in an experiment, Nasser Alamri selected 6 different file types (pdf, txt, jpg, doc, html, and exe) and reduced the features via the PCA feature reduction. Then and there, he compared the reduced features set with 5 algorithms of SVM, KNN, the neural network based on the core function radius, the neural network with perceptron core, and linear discriminant analysis on the same database. The relevant database has randomly extracted the sample data from the Govdoc dataset, and 100 samples were taken from each file of which the subsets are also randomly extracted. The results are shown in Table 8. We also matched a variety of file types with the files provided to compare our work with other research.

Table 8. Results obtained in Alamri's 2015 paper

classifie r	Number of features	fragm ent	Train/te st	accuracy
LDA	64	500 Byte	90/10	93%
SVM	64	500 Byte	80/20	94%
K-NN	8	500 Byte	90/10	97%
NN-RBF	4	1000 Byte	80/20	88%
NN-MLP	64	500 Byte	90/10	94%

As shown in Table 8, the KNN algorithms with the accuracy of 97% and the NN-RBF algorithm with an accuracy of 88% have the least accuracy in the Nasser Alamri paper. Figure 5 shows the comparison between our proposed research algorithms and the Alamri's paper.

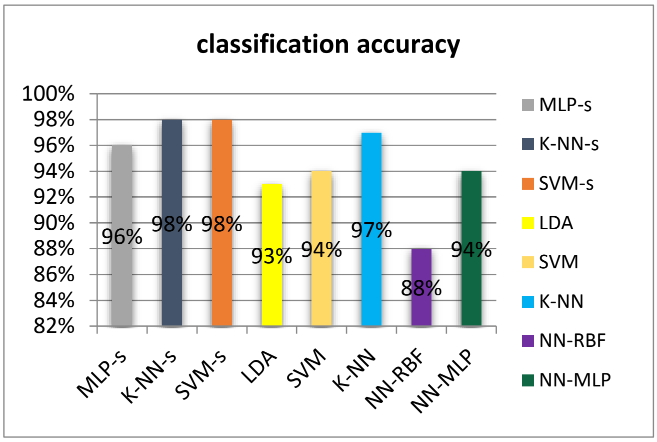

Fig.5. Comparison of the proposed algorithm of this research with the results of Alamri's paper (2015)

In Figure 5, the MLP-s column describes the MLP classification algorithm by SFFS feature reduction approach, and the K-NN-s column represents the K-NN classification algorithm by the same approach. Further, the SVM-s column represents the SVM classification algorithm by SFFS feature reduction approach.

According to Figure 5, the KNN-s, SVM, and MLP-s approaches by respectively 1%, 4%, and 2% increase in the accuracy rate show the increasing trend in the accuracy of this research compared to Alamri's. Also, KNN and SVM algorithms combined with the SFFS feature reduction approach indicate the highest accuracy of the categorization (98%) among the eight algorithms examined.

7. Conclusion

File type's detection is an essential task for many security programs. Although there are lots of programs to deal with detection of computer file types, there are just minimal algorithms for detecting them. However, the primary issue in detecting the file type is the classification of the file fragments since there are no headers (a part of the file containing information about the file type) or systemic file information, which can specify the file type. The general algorithm to classify file fragments is to examine the histogram of its byte frequency and sometimes analyze other statistics obtained. The statistical distance between the histogram and the known distributions of different files types can be calculated, which will be used to distinguish different data types. Based on recent research, although the classification of file fragments in many common file types can be done with high accuracy, this algorithm has some limitations to detect the type of file, running time and accuracy. A higher degree of accuracy obtained in this study compared with previous studies.

In this paper, the problem of recognizing the file fragment was begun by considering 1000-byte and 500-byte fragments of each file. The BFD algorithm extracted the features of each file fragment. Then, by two SFS and SFFS feature reduction algorithms, the features extracted from each fragment were reduced to 24-39 features depending on the length of the file fragment. The reduced features were considered as inputs of three MLP, KNN, and SVM classification algorithms to obtain the accuracy of the classification algorithms. The best result in this study was achieved as 98%

References

1. McDaniel M, Heydari M H. Content-based file type detection algorithms. In Proc. the 36th IEEE Annual Hawaii International Conference. System Science, Jan 2003, pp.10.
2. Li W, Wang K, Stolfo S J, Herzog B. Fileprints: Recognizing file types by n-gram Analysis. In Proc. the 6th IEEE Systems. Man and Cybernetics Information Assurance Workshop, West Point, New York, Jane 2005, pp.64-71.
3. Mabodi, K., Yusefi, M., Zandiyan, S., Irankhah, L., & Fotohi, R. Multi-level trust-based intelligence schema for securing of internet of things (IoT) against security threats using cryptographic authentication. The Journal of Supercomputing, 1-25.
4. Fotohi, R., Jamali, S., Sarkohaki, F., & Behzad, S. (2013). An Improvement over AODV routing protocol by limiting visited hop count. International Journal of Information Technology and Computer Science (IJITCS), 5(9), 87-93.
5. Zandiyan S, Fotohi R, Koravand M. P-method: Improving AODV routing protocol for against network layer attacks in mobile Ad-Hoc networks. International Journal of Computer Science and Information Security. 2016 Jun 1;14(6):95.
6. Jamali, S., & Fotohi, R. (2017). DAWA: Defending against wormhole attack in MANETs by using fuzzy logic and artificial immune system. the Journal of Supercomputing, 73(12), 5173-5196.
7. Lodeiro-Santiago, M., Caballero-Gil, P., Aguasca-Colomo, R., & Caballero-Gil, C. (2019). Secure UAV-Based System to Detect Small Boats Using Neural Networks. Complexity, 2019.
8. Fotohi, R., Heydari, R., & Jamali, S. (2016). A Hybrid routing method for mobile ad-hoc networks. Journal of Advances in Computer Research, 7(3), 93-103.
9. Fotohi, R., & Bari, S. F. (2020). A novel countermeasure technique to protect WSN against denial-of sleep attacks using firefly and Hopfield neural network (HNN) algorithms. The Journal of Supercomputing, 1-27.
10. Karresand M, Shahmehri N. File Type Identification of Data Fragments by Their Binary Structure. In Proc. the IEEE Workshop. Information Assurance, July 2006, pp.140-147.
11. Veenman Core J. Statistical disk cluster classification for file carving.In Proc. The 3rd IEEE international symposium. information assurance and security, Aug 2007, pp. 393-398.
12. Behzad, S., Fotohi, R., & Dadgar, F. (2015). Defense against the attacks of the black hole, gray hole and wormhole in MANETs based on RTT and PFT. International Journal of Computer Science and Network Solutions (IJCSNS), 3, 89-103.
13. Seyedi, B., & Fotohi, R. NIASHPT: a novel intelligent agent-based strategy using hello packet table (HPT) function for trust Internet of Things. The Journal of Supercomputing, 1-24.
14. Fotohi, R., Bari, S. F., & Yusefi, M. (2019). Securing Wireless Sensor Networks Against Denial-of-Sleep Attacks Using RSA Cryptography Algorithm and Interlock Protocol. International Journal of Communication Systems.
15. Sarkohaki, F., Fotohi, R., & Ashrafian, V. (2017). An efficient routing protocol in mobile ad-hoc networks by using artificial immune system. International Journal of Advanced Computer Science and Applications (IJACSA), 8 (4).
16. Fotohi, R., Ebazadeh, Y., & Geshlag, M. S. (2016). A new approach for improvement security against DoS attacks in vehicular ad-hoc network. International Journal of Advanced Computer Science and Applications, 7(7), 10-16.
17. Behzad, S., Fotohi, R., Balov, J. H., & Rabipour, M. J. (2018). An Artificial Immune Based Approach for Detection and Isolation Misbehavior Attacks in Wireless Networks. JCP, 13(6), 705-720.
18. Calhoun W, Coles D. Predicting the types of file fragments. Journal Digital Investigation: The International Journal of Digital Forensics & Incident, 2008, 5: S14-S20.
19. Sportiello L, Stefano Z. Context-Based File Block Classification. Advances in Digital Forensics VIII, Springer Berlin Heidelberg, 2012, pp. 67-82.
20. Gopal S, Yang Y, Salomatin k, Carbonell j. Statistical learning for file-type recognition. In Proc. The 10th IEEE International Conference . Machine Learning and Applications and Workshops (ICMLA), 2011, vol. 1, pp. 68-73.
21. Fotohi, R. (2020). Securing of Unmanned Aerial Systems (UAS) against security threats using human immune system. Reliability Engineering & System Safety, 193, 106675.
22. Fotohi, R.; Nazemi, E. An Agent-Based Self-Protective Method to Secure Communication between UAVs in Unmanned Aerial Vehicle Networks. Preprints 2020, 2020010229 (doi: 10.20944/preprints202001.0229.v1).
23. Behzad, S., Fotohi, R., & Jamali, S. (2013). Improvement over the OLSR routing protocol in mobile Ad Hoc networks by eliminating the unnecessary loops. International Journal of Information Technology and Computer Science (IJITCS), 5(6), 2013.

doi:10.20944/preprints202001.0305.v1

24. Erbacher R F, Moody S J. Sadi-statistical analysis for data type recognition. In Proc. The 3rd International Workshop. Systematic Approaches to Digital Forensic Engineering, May 2008, pp.41-54.

25. Dunham J G, Sun M T, Tseng J. Classifying File Type of Stream Ciphers in DepthUsing Neural Networks. In Proc. The The 3rd ACS/IEEE International Conference. Computer Systems and Applications, Jan 2005, pp.97.

26. Like Z, White G B. An Approach to Detect Executable Content for Anomaly Based Network Intrusion Detection. In Proc. IEEE International Conference. Parallell and Distributed processing Symposium, IPDPS, Long Beach, California, 2007, pp.1-8.

27. Cao D, Luo J, Yin M, Yang H. Feature selection based file type recognition algorithm. In Proc. IEEE International Conference. Intelligent Computing and Intelligent Systems, (ICIS), 2010,vol 3, pp.58-62.

28. Ahmed I, Lhee K, Shin H J, Hong M. Content-based file-type recognition using cosine similarity and a divide-and-conquer approach. IETE Technical Review, 2010, vol. 27(6), pp. 465-477.

29. Ahmed I, Lhee K, Shin H J, Hong M P. Fast Content-Based File Type Recognition. In Proc. The 7th Annual IFIP .Advances in Digital Forensics VII, Orlando, FL, USA, Springer Berlin Heidelberg, 2011, pp. 65-75.

30. Alamri N S, Allen W H. A Comparative Study of File-Type Recognition Techniques. In Proc. Proceedings of the IEEE SoutheastCon, Fort Lauderdale, Florida, April 2015, pp.1-5.

31. Wayne Whitney A. A Direct algorithm of Nonparametric Measurement Selection. IEEE Transaction on Computers, 1971, pp. 1100-1103.

32. Pudill p, Novovi j, Kittler J. Floating search algorithms in feature selection. Department of Electronic and Electrical Engineering, ElSEVIER, 1994, pp. 1119-1125.

33. Kulkarni S, Harman G. An elementary introduction to statistical learning theory. John Wiley & Sons, 2011.

34. Fotohi, R., & Jamali, S. (2014). A comprehensive study on defence against wormhole attack methods in mobile Ad hoc networks. International journal of Computer Science & Network Solutions, 2, 37-56.

35. Jamali, S., & Fotohi, R. (2016). Defending against wormhole attack in MANET using an artificial immune system. New Review of Information Networking, 21(2), 79-100.

36. Jamali, S., Fotohi, R., Analoui, M. (2018). An Artificial Immune System based Method for Defense against Wormhole Attack in Mobile Adhoc Networks. TABRIZ JOURNAL OF ELECTRICAL ENGINEERING, 47(4), 1407-1419