Karbala International Journal of Modern Science

Volume 8 | Issue 2 Article 6

A Content-based File Identification Dataset: collection, construction, and evaluation

Saja Dheyaa Khudhur
Computer Engineering Department, University of Technology -Iraq, 120099@uotechnology.edu.iq

Hassan Awheed Jeiad
Computer Engineering Department, University of Technology -Iraq

Recommended Citation
Khudhur, Saja Dheyaa and Jeiad, Hassan Awheed (2022) "A Content-based File Identification Dataset: collection, construction, and evaluation," *Karbala International Journal of Modern Science*: Vol. 8 : Iss. 2 , Article 6.
Available at: https://doi.org/10.33640/2405-609X.3222

A Content-based File Identification Dataset: collection, construction, and evaluation

Abstract

File-Type Identification (FTI) is one of the essential functions that can be performed by examining the data blocks' magic numbers. However, this examination leads to a challenge when a file is corrupt, or these magic numbers are missing. Content-based analytics is the best way for file type identification when the magic numbers are not available. This paper prepares and presents a content-based dataset for eight common types of files based on twelve features. We designed our dataset to be used for supervised and unsupervised machine learning models. It provides the ability to classify and cluster these types into two levels, as a fine-grain level (by their file type exactly, JPG, PNG, HTML, TXT, MP4, M4A, MOV, and MP3) and as a coarse-grain level (by their broad type, image, text, audio, video). A dataset quality and features assessments are performed in this study. The obtained results show that our dataset is high-quality, non-biased, complete, and with an acceptable duplication ratio. In addition, several multi-class classifiers are learned by our data, and classification accuracy of up to 81.8% is obtained. The main contributions of this work are summarized in constructing a new publicly available dataset based on statistical and information content-related features with detailed assessments and evaluation.

Keywords

Dataset; File-Type Identification (FTI); File Type classification; Fragment File Type Identification (FFTI); Machine Learning; Feature extraction; Dataset Evaluation; content-based analysis.

RESEARCH PAPER

A Content-based File Identification Dataset: Collection, Construction, and Evaluation

Saja Dheyaa Khudhur*, Hassan Awheed Jeiad

Computer Engineering Department, University of Technology, Iraq

Abstract

File-Type Identification (FTI) is one of the essential functions that can be performed by examining the data blocks' magic numbers. However, this examination leads to a challenge when a file is corrupt, or these magic numbers are missing. Content-based analytics is the best way for file type identification when the magic numbers are not available. This paper prepares and presents a content-based dataset for eight common types of files based on twelve features. We designed our dataset to be used for supervised and unsupervised machine learning models. It provides the ability to classify and cluster these types into two levels, as a fine-grain level (by their file type exactly, JPG, PNG, HTML, TXT, MP4, M4A, MOV, and MP3) and as a coarse-grain level (by their broad type, image, text, audio, video). A dataset quality and features assessments are performed in this study. The obtained results show that our dataset is high-quality, non-biased, complete, and with an acceptable duplication ratio. In addition, several multi-class classifiers are learned by our data, and classification accuracy of up to 81.8% is obtained. The main contributions of this work are summarized in constructing a new publicly available dataset based on statistical and information content-related features with detailed assessments and evaluation.

Keywords: Dataset, File-type identification (FTI), File type classification, Fragment file type identification (FFTI), Machine learning, Feature extraction, Dataset evaluation, Content-based analysis

1. Introduction

File-Type Identification (FTI) is one of the essential processes of the operating system by which the computer chooses the program to process a file [1]. A variety of commercial software is available for this process, aka Commercial-Off-The-Shelf (COTS), such as TrID, Libmagic, DROID, Outside-In. Through that software, the file type identification can be done through examination of the data blocks' magic numbers (e.g., file footers and signatures), file system metadata, file extensions, or packet header information. However, this method that relies on file signatures designed to recognize its type becomes helpless when corruption or missing in data happens. Here, in such cases, content-based analytics is the best method for file type identification [2]. This analytics comprises three approaches, semantic parsing, non-semantic parsing, and Machine Learning (ML). Semantic parsing depends on formal representations of linguistic meaning, natural-language structures, and data structure and logic. So, the main utility limitation of this approach is the reality that such representations and structures are not predominate in many types of files.

Moreover, non-semantic parsing includes searching for standard strings to specific file types, e.g., the 'endstream', and 'obj<</', '>>stream' are the standard strings for PDF files. But the utility limitation here is because not all types of data apply to this rule, such as TXT files. So, the ML approach provides the optimal way for data type classification for many data types due to its statistical classification capabilities [3].

Long ago, several statistical [4], rule-based, or ML approaches [5–11] were presented for file type

Received 2 November 2021; revised 25 January 2022; accepted 29 January 2022.
Available online 1 May 2022

* Corresponding author at:.
E-mail addresses: 120099@uotechnology.edu.iq (S.D. Khudhur), 120004@uotechnology.edu.iq (H.A. Jeiad).

https://doi.org/10.33640/2405-609X.3222

classification. The supervised and unsupervised ML were utilized for the file type classification and clustering, respectively. With the ML models, the identification process is based on the file contents regardless of the signature or other file magic characteristics [12].

In 2003 the first file identification approaches based on no-header and no-footer had appeared [13]. After that, several approaches were presented to build predictive models for each file type, Li et al. [14] and Karresand [15] adopted the k-means clustering algorithm. Conti et al. [6] and Axelsson [5] applied the k-Nearest Neighbour (kNN) approach. Fitzgerald et al. [16], Zheng et al. [17], and [12] utilized Support Vector Machine (SVM) classifier. More recently, researchers in the field of FTI are taking more interest in the ML approach due to its importance in many fields, such as digital forensics [18], information security, anti-virus [19], data carving, cybercrime issues, and big data applications [20,21].

All the presented ML approaches were trained and tested by the featured dataset. Unfortunately, the authors built almost all the featured datasets for their specified models without assessing their data validity or providing it as public for free access.

We reviewed research related to FTI or File Fragment Type Identification (FFTI). Other than searching the search engine of high-impact research data repositories, such as the UCI, figshare, Mendeley Data, OSF Home, and IEEE DataPort, there is no ready-made dataset for machine learning in this field of identification. Instead, there are FFTI datasets available that contain file fragments with different byte sizes. For example, Govind et al. shared file fragments from 75 popular file types in different granularities with six scenarios [22]. Reyhane et al. presented a dataset based on ten image file formats that contain 25,600 file fragments as a total with different compression settings [23]. In Ref. [24], Fatemeh, et al. offered a file fragments dataset for five textual file formats in three other languages. Moreover [25], presented a file fragments dataset that contains 600,000 fragments based on ten video file formats with different video codec types.

In general, all these datasets are composed of file fragments of the corresponding types. Therefore, these fragment datasets are suitable for deep learning algorithms. But due to the ML requirements, a numerical featured dataset must be used in the learning process, so these fragment datasets need features extractions and data preprocessing to be suitable to the ML models.

In this paper, we construct and present a content-based FTI dataset that comprises twelve features for eight common types of files (JPG, PNG, HTML, TXT, MP4, M4A, MOV, and MP3). The constructed dataset is developed for supervised and unsupervised ML models. It provides the ability to classify and cluster the types as mentioned earlier into two levels, at a fine grain level (by their file type exactly, JPG, PNG, HTML, TXT, MP4, M4A, MOV, and MP3) and at a coarse-grain level (by their broad type, image, text, audio, video). The contributions of our work include:

- A novel machine learning-based dataset (8 popular file types with 12 high impact statistical features) is publicly available at https://dx.doi.org/10.17605/OSF.IO/8BK3R
- A detailed evaluation and assessment for the constructed dataset are illustrated.

The paper is structured as follows. Section 2 presents the construction procedure of the dataset. Then, in Section 3, we describe and deliberate the results of the evaluation process for the dataset and the extracted features. Finally, Section 4 discusses the conclusion of this research.

2. Dataset construction

Basically, the construction of the proposed dataset contains two stages, data collection and features extractions.

2.1. Data collection

The development of supervised ML models required a balanced, labelled dataset that has been used in training and testing the intended machine learning models. In this step, we constructed a dataset with selective features with a high impact factor in content-based FTI and FFTI models. The constructed dataset was collected manually from a Garfinkel file corpus [26]. This publicly available Govdocs1 corpus was utilized in many works for FTI and FFTI [12,16,27−31]. Due to the limitations of this group in the other file types, as it lacks video and audio files, and because the goal of this research is to classify files into the four previously mentioned categories, our data collection is extended by a file corpus created by Portaz et al. [32]. Their publicly available GaRoFou corpus was utilized. This corpus comprised 96 minutes of video collected from the Museum of Fourvière in Lyon. Additionally, audio files were collected manually from the Pixabay website and our private database.

The basic elements of our constructed dataset are file fragments. The procedure of constructing the basic elements is shown in Fig. 1 and described as follows:

I. Initially, a files pool was formed from the files that would be fragmented. However, due to our goals of using a file fragment of size 512 bytes that does not include the header signature, we excluded the first 512 bytes from each file. To achieve these goals, the files with insufficient size were filtered out, and the smallest size allowed for the files was 1024 bytes.

II. All files are partitioned into fragments of size 512-bytes. In fact, this size was adopted because it is the most popular fragment size used by the previous works [3,5,12,16,30,31]. The reason behind that size was that it is the smallest size of hard drive sectors. Despite that fact, Penrose et al. [29] used a fragment of size 4096-bytes. They have argued that that size is the safest choice, being the approved hard drive size for most manufacturers since 2011 [29], but Axelsson noted that the 512-byte size is a conservative choice [5].

III. Excluding the first 512 bytes from each file prevents the models from being skewed by header data. Moreover, to maintain a fixed fragment size, the last fragment was excluded if it was shorter than 512-bytes.

In total, our dataset is composed of 177,951 fragments for eight file types, as illustrated in Table 1.

2.2. Features extraction

From the data collection, statistical computation techniques and information content-related features are extracted as follows.

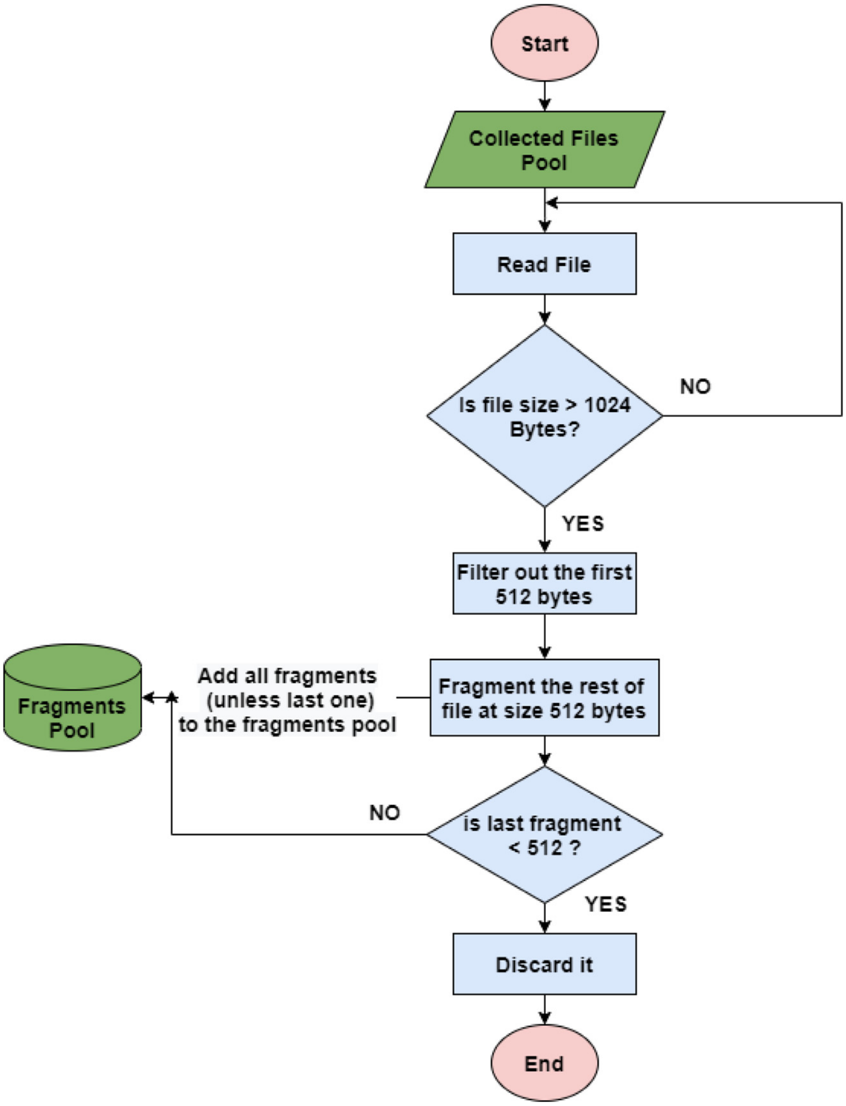
Start
Collected Files Pool
Read File
Is file size > 1024 Bytes?
NO
YES
Filter out the first 512 bytes
Fragment the rest of file at size 512 bytes
Add all fragments (unless last one) to the fragments pool
Fragments Pool
is last fragment < 512 ?
NO
YES
Discard it
End

Fig. 1. Fragmentation producer flowchart.

Table 1. Illustration of the dataset.

Class No.	File Type	Number of file fragments
1	**Image Data**	**33,029**
	JPG	12,675
	PNG	20,354
2	**Text Data**	**25,999**
	HTML	13,096
	TXT	12,903
3	**Video Data**	**86,027**
	MP4	66,027
	MOV	20,000
4	**Audio Data**	**32,896**
	M4A	13,200
	MP3	19,776
Total		**178,031**

a) Byte Frequency Distribution (BFD)

BFD is the common feature that is primarily used in FTI, such as unigram frequencies, i.e., the occurrence frequency of each byte value in the file. Moreover, the bigram and trigram frequencies are other types of BFD that are unique from others in the concept of consecutive bytes length.

Since the unigram deals with the byte level, such BFD produces 256 different values [33]. Therefore, the set of normal distribution (mean absolute deviation and standard deviation) and mean value are also fitted into our feature space and are extracted from each 512-byte fragment.

Additionally, the probability distribution of the bytes at the unigram level is another feature of our space. It is computed simply by dividing the occurrence frequency of each byte value in the fragment, taken from the unigram vector, by the fragment size. After that, and for the reason of the curse of dimensionality, the normal distribution (mean absolute deviation and standard deviation) and mean value of the probability distribution vector are taken.

b) Mean Byte Value

Another feature based on the content is the Mean Byte Value, i.e., the arithmetic means of the byte values in a fragment, which is computed simply by summing the byte values in a fragment and dividing the sum by the fragment sizes. It is calculated as:

$$\mu = \frac{1}{n}\sum_{i=0}^{n} B_i \tag{1}$$

where B_i is the byte values (in decimal format) and n is the fragment size in bytes. In our feature space, the value of n is 512.

c) Shannon Entropy

Furthermore, the Shannon entropy, a complexity and information content-related feature for the whole fragment, is computed based on the mean byte value and used as another dimension in our feature space. Shannon entropy is an established technique developed by Claude Shannon [34] for measuring uncertainty. In other words, it measures the amount of randomness or disorder in a segment in terms of the number of bits per sample. It is computed as [35]:

$$E(x) = -\sum_{i=0}^{n} p(x_i)\log_2 p(x_i) \tag{2}$$

where *x* is a vector of data symbols with a length of n for each of these symbols, in byte-level analysis, which is the state of consideration in our feature space, *x* is composed of 256 symbols of length 8 bits. While *p(x)* is a probability mass function that, in our feature space, denotes the probability of occurrence of byte value *i* in the fragment.

The entropy values in byte-level analysis, ranging from 0 to $\log_2 2^8$, provide a quick and convenient method for analysing files at the binary level, which in turn offers a possible pre-processing step for identifying suspicious regions in a file. Furthermore, these regions can be analysed with reverse-engineering disassembling tools [36]. Moreover, the entropy analysis can be used to distinguish the file type at the general level (text, image, video, audio) and the construction level (binary code, compressed data).

d) Hamming Weight

Hamming weight is the ratio of the total number of set bits to the total number of bits in a segment. It has a significant role in several disciplines, including coding theory, information theory, and cryptography. In byte-level analysis, as used in this paper, each byte of a given fragment is converted to a binary representation (8-bit for each byte, composed of zeros and ones). Then the total number of ones is divided by the fragment size in a bit (i.e., 512*8). As illustrated in the equation below:

$$H(x) = \frac{\text{total no. of ones}}{\text{fragment size in a bit (total number of bits)}} \tag{3}$$

where x is a binary segment that, in our features space, is the fragment of size 512-bytes that was represented in binary format.

e) Longest Streak

It refers to the longest continuous iterations for a byte in a file. The value of the byte that is contained in this longest streak is also a feature in our space named the ‘Longest Byte’.

As a result, the constructed dataset will be classified into two classes, the main class and a subclass. In the main class, all features are distributed, with a coarse-grain, over four classes based on broad classification (video data, image data, audio data, and text data). While in the sub-main class, all features are distributed (with a fine-grain) over eight classes based on the selected file types. Table 2 shows the details of the constructed dataset.

3. Dataset evaluation

First, the data represent real-world objects where each column is a dataset of the dataset, and each cell is the value that that dataset acquires for a row. In this section, data quality and features assessments are checked.

3.1. Data quality assessment

To ensure the value of the data and ensure producing high-quality results from it, three critical features are argued by Ref. [37] that must be examined, which are *quality, quantity*, and *availability* of data. A data quality test is essential in dataset evaluation [38]. A quantity test indicates whether there is enough data to train and test the models, which, for example, is necessary for ML models. Moreover, data availability is critical because it allows other researchers to exploit that data and reproduce potentially improved results [37].

A data quality check encompasses multiple dimensions. The common ones are accuracy, uniqueness, completeness, validity, and consistency. In this section, the primary qualities of the constructed dataset are checked using the Streamlit Python application based on the following metrics:

a. Missing Values: It measures the number of missing/null values that, if any, would cause a bias in the data that in turn drive vague results.
b. Completeness Ratio: This is the number of non-missing values records divided by the total number of those in the dataset. In practice, 85% is considered to be an acceptable ratio of completeness.
c. Duplication Rate: This is the ratio of duplicate records to the total number of those.
d. Normality: This is the distribution metric of the dataset. It determines how far the dataset is from the normal distribution. This test uses the skewness technique where this technique is a measure of asymmetry in the distribution.

Where the quantity criteria are tested in section (3.3), the ability for training and testing for multi-ML models are examined. Besides the availability criteria, the dataset is already publicly available at https://dx.doi.org/10.17605/OSF.IO/8BK3R.

3.2. Features assessment

In ML applications, identifying the dataset's independent variables (features) is significant in model building. In this process, the features that have the most influence on the outcomes of an ML model are identified. This process depends on the properties of the features [39].

In this stage, we assess the extracted features regardless of the ML algorithms as follows.

1. Finding the relationship between the independent variables, i.e., the correlation between the features.

Table 2. Baseline details of the constructed dataset.

Feature Index	Feature Name	Total Samples	Main Classes	Sub-classes
F1	Mean Byte Value	178,031	Image	JPG
F2	Probability Distribution (STD)			PNG
F3	Probability Distribution (Mean)			
F4	Probability Distribution (MAD)		Text	HTML
F5	Longest Streak			TXT
F6	Longest Byte			
F7	Unigram Frequencies (STD)		Video	MP4
F8	Unigram Frequencies (Mean)			MOV
F9	Unigram Frequencies (MAD)			
F10	Hamming Weight		Audio	MP3
F11	Shannon Entropy			M4A
F12	Unigram frequencies Vector			

Table 3. Our dataset quality assessment test results.

Assessment test	Results	Discription
Missing Values	0	Not- biased
Completeness ratio	100%	completed
duplication rate	5%	Acceptable
Normality	4.0498	Right-skewed

2. Calculating the severity of multicollinearity in an ordinary least square regression analysis for the independent features. That is done by checking the independent variables' Variance Inflation Factor (VIF).

3.3. *Analysis of the data evaluations results*

This section illustrates and deliberates the assessment results for dataset quality and the extracted features mentioned in sections (3.1) and (3.2). Table 3 illustrates the results of our dataset quality assessment test.

From that assessment results, we note that our dataset is unbiased because it does not contain missing values, which indicates the reliability and trustworthiness of the dataset. However, missing data may lose crucial values that impact the model's performance because training an ML model with a biased dataset can drastically affect the model's quality.

Since the completeness ratio test depends mainly on the missing values test, our dataset is complete. Thus, it has no gaps in it, and as stated above, that dataset is reliable and qualified for decision making.

Table 4. VIF among the independent variables.

Feature index	VIF Factor
F1	368.4103
F2	2218918
F3	3.06E+08
F4	13024942
F5	79.98515
F6	2.883988
F7	2218905
F8	3.06E+08
F9	13024693
F10	148.8902
F11	242.0534

Testing the dataset shows that it has a duplication rate equal to 0.05. This ratio was commonly accepted and had no crucial impact on the data quality. However, the normality test that is performed on our dataset shows that it has a right-skewed ratio of 4.0498. This type of skewness means that the distribution has a long tail that extends to the right of the x-axis. This tail region may, in some cases, act as an outlier. These outliers adversely affect some statistical models' performance, especially the regression-based ones. However, tree-based models are robust to the outlier.

Regarding features assessment, the relationship results among the independent variables are formed in a correlation heatmap that visually depicts the relationship between the variables in Fig. 2. Also, the VIF result listed in Table 4 provides an index of the variance measurement (the square of the estimate's standard deviation) for all the selected features.

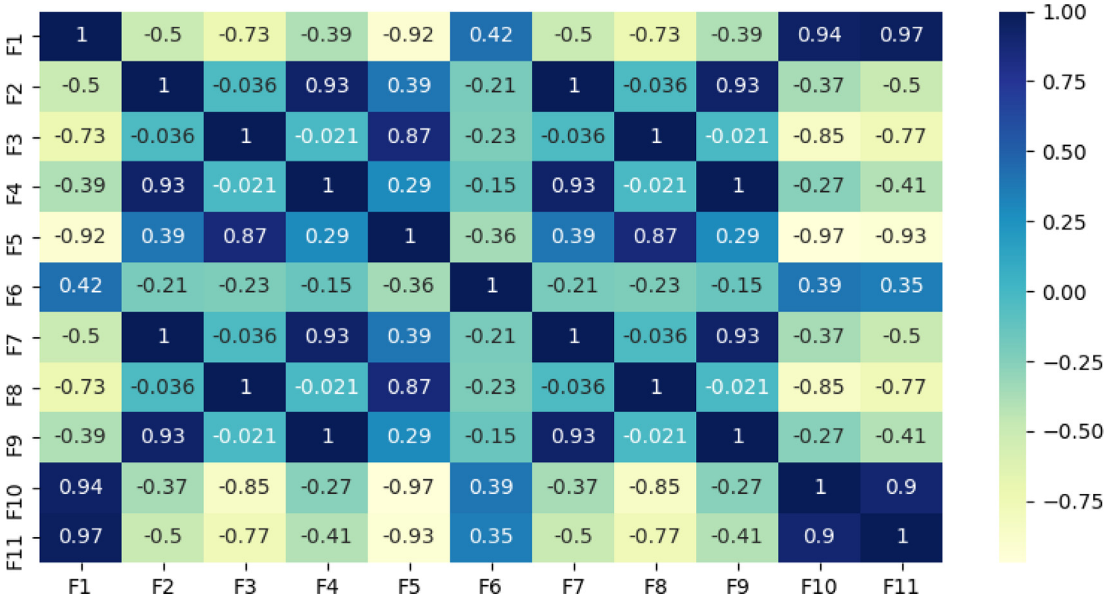

Fig. 2. Correlation heatmap.

Table 5. Classification accuracy.

Type of Model	Classifier	Features	Testing Accuracy
Tree-based	Classification Tree	F1−F11	75%
	Random Forest		81.8%
Non-tree-based	Logistic Regression		62.8%
	K Nearest Neighbours		73%

As illustrated from Fig. 2, some features have high correlations with each other, for example, F2, F3, and F4 with F7, F8, and F9, respectively. In some cases, for instance, in the age of big data, reducing the number of features through selecting the more crucial information has a significant role in building high efficiency and less complex models. That reduction, in turn, will save time and resources. Therefore, it is possible to exclude one of the associated features, which may not affect the accuracy negatively. Generally, the inclusion and exclusion of the dataset's features depend on the ML algorithm, learning type, problem, and target.

The study of identifying the file type includes a different dataset and many file types, which complicates the comparison with other research. Moreover, no publicly available dataset can be compared with our dataset, so we evaluate the dataset capabilities by training and testing the ML models. First, the dataset is divided into training/testing data at an 80:20 ratio, fitting those to the ML algorithms. Then, to support our claim for the skewness effect, tree-based and non-tree-based algorithms from the scikit learn models are utilized [40]. These algorithms are trained by the first eleven features and tested for classifying fragments to their main class. Table 5 illustrates the testing accuracy for the adopted models. The accuracy results of the models are calculated based on the default hyperparameter. Thus, that might make the calculation composed of some deviation's consequent to the data's random distribution. A suit of reasonable default hyperparameters was implemented for the sci-kit learn models. However, not all of those are optimally ensured to any problem. However, multiple processes should be performed to achieve a better accuracy-based model. Some of those are model-hyperparameters tuning, dataset pre-processing, and the feature engineering process, where each one is based on model, target, and problem.

4. Conclusion

Recognizing the type of file is a crucial job for many applications. Although many tools deal with recognizing computer file types, there is still the need for algorithms that detect them. Moreover, the classification of file fragments is the primary issue since there are no headers or systemic information for a file that can identify the file and fragment type. The content-based analysis is an attractive algorithm for classifying the type of file., which examines and analyses the byte frequency attributes and other statistics patterns. This paper produces a novel high-quality dataset with twelve crucial features, non-biased, complete, and has an acceptable duplication ratio. The obtained dataset is right-skewed, and from the obtained classification results, this skewness is not highly impacting the tree-based model. So, if a regression-based model or non-tree-based models are to be learned with our dataset, it is necessary to transform the skewed data to close enough to a normal or Gaussian distribution. The common transformations method includes logarithmic, square root, and reciprocal.

Moreover, dropping the outliers also helps normalize the skewed dataset. As a result, greater accuracy was obtained in this study from the tree-based models. The models were learned with a set of default hyperparameters that make achieving a better accuracy highly likely.

References

[1] J. Sester, D. Hayes, M. Scanlon, N.-A. Le-Khac, A comparative study of support vector machine and neural networks for file type identification using n-gram analysis, Forensic Sci Int: Digit Invest 36 (2021) 301121, https://doi.org/10.1016/j.fsidi.2021.301121.

[2] K. Bhat, J.T. Lam, F. Zulkernine, Content-based file type identification, in: 2018 10th international conference on electrical and computer engineering, (ICECE), 2018, pp. 277−280, https://doi.org/10.1109/ICECE.2018.8636693.

[3] N.L. Beebe, L.A. Maddox, L. Liu, M. Sun, Sceadan: using concatenated N-gram vectors for improved file and data type classification, IEEE Trans Inf Forensics Secur 8 (2013) 1519−1530, https://doi.org/10.1109/TIFS.2013.2274728.

[4] V. Roussev, C. Quates, File fragment encoding classification—an empirical approach, Digit Invest 10 (2013) S69−S77, https://doi.org/10.1016/j.diin.2013.06.008.

[5] S. Axelsson, The Normalised Compression Distance as a file fragment classifier, Digit Invest 7 (2010), https://doi.org/10.1016/j.diin.2010.05.004. S24−S31.

[6] G. Conti, S. Bratus, A. Shubina, B. Sangster, R. Ragsdale, M. Supan, et al., Automated mapping of large binary objects using primitive fragment type classification, Digit Invest 7 (2010), https://doi.org/10.1016/j.diin.2010.05.002. S3−S12.

[7] I. Ahmed, K.-S. Lhee, Detection of malcodes by packet classification, in: 2008 Third international conference on availability, Reliability and Security, 2008, pp. 1028−1035, https://doi.org/10.1109/ARES.2008.100.

[8] I. Ahmed, K.-S. Lhee, H. Shin, M. Hong, On improving the accuracy and performance of content-based file type identification, in: Information security and privacy, Springer Berlin Heidelberg, 2009, pp. 44−59, https://doi.org/10.1007/978-3-642-02620-1_4.

[9] I. Ahmed, K.-S. Lhee, H. Shin, M. Hong, Fast file-type identification, in: Proceedings of the 2010 ACM symposium

on applied computing, Association for Computing Machinery, New York, NY, USA, 2010, pp. 1601−1602, https://doi.org/10.1145/1774088.1774431.

[10] I. Ahmed, K.-S. Lhee, H.-J. Shin, M.-P. Hong, Fast content-based file type identification, in: Advances in digital forensics VII, Springer Berlin Heidelberg, 2011, pp. 65−75, https://doi.org/10.1007/978-3-642-24212-0_5.

[11] I. Ahmed, K.-S. Lhee, Classification of packet contents for malware detection, J Comput Virol 7 (2011) 279, https://doi.org/10.1007/s11416-011-0156-6.

[12] M. Bhatt, A. Mishra, M.W.U. Kabir, S.E. Blake-Gatto, R. Rajendra, M.T. Hoque, et al., Hierarchy-based file fragment classification, Mach Learn Knowl Extract 2 (2020) 216−232, https://doi.org/10.3390/make2030012.

[13] M. McDaniel, M.H. Heydari, Content based file type detection algorithms, in: 36th annual Hawaii international conference on system sciences, 2003, p. 10, https://doi.org/10.1109/HICSS.2003.1174905. Proceedings of the, 2003.

[14] W.-J. Li, K. Wang, S.J. Stolfo, B. Herzog, Fileprints, Identifying file types by n-gram analysis, in: Proceedings from the sixth annual IEEE SMC information assurance workshop, 2005, pp. 64−71, https://doi.org/10.1109/IAW.2005.1495935.

[15] Shahmehri Karresand, File type identification of data fragments by their binary structure, IEEE Information Assurance Workshop, 2006, pp. 140−147, https://doi.org/10.1109/IAW.2006.1652088.

[16] S. Fitzgerald, G. Mathews, C. Morris, O. Zhulyn, Using NLP techniques for file fragment classification, Digit Invest 9 (2012), https://doi.org/10.1016/j.diin.2012.05.008. S44−S49.

[17] F.K. Nakano, W.J. Pinto, G.L. Pappa, R. Cerri, Top-down strategies for hierarchical classification of transposable elements with neural networks, in: 2017 International joint conference on neural networks (IJCNN), 2017, pp. 2539−2546, https://doi.org/10.1109/IJCNN.2017.7966165.

[18] M.A. Neaimi, H.A. Hamadi, C.Y. Yeun, M.J. Zemerly, Digital forensic analysis of files using deep learning, in: 2020 3rd international conference on signal processing and information security, ICSPIS), 2020, pp. 1−4, https://doi.org/10.1109/ICSPIS51252.2020.9340141.

[19] D. Gibert, C. Mateu, J. Planes, The rise of machine learning for detection and classification of malware: research developments, trends and challenges, J Netw Comput Appl 153 (2020) 102526, https://doi.org/10.1016/j.jnca.2019.102526.

[20] A. Bhat, A. Likhite, S. Chavan, L. Ragha, File fragment classification using content based analysis, ITM Web Conf. 40 (2021), 03025, https://doi.org/10.1051/itmconf/20214003025.

[21] M. Masoumi, A. Keshavarz, R. Fotohi, File fragment recognition based on content and statistical features, Multimed Tool Appl 80 (2021), https://doi.org/10.1007/s11042-021-10681-x, 18859−18874.

[22] G. Mittal, P. Korus, N. Memon, File fragment type (FFT) - 75 dataset, 2019, https://doi.org/10.21227/KFXW-8084.

[23] R. Fakouri, M. Teimouri, Dataset for file fragment classification of image file formats, BMC Res Notes 12 (2019) 774, https://doi.org/10.1186/s13104-019-4812-0.

[24] F. Mansouri Hanis, M. Teimouri, Dataset for file fragment classification of textual file formats, BMC Res Notes 12 (2019) 801, https://doi.org/10.1186/s13104-019-4837-4.

[25] N. Sadeghi, M. Fahiminia, M. Teimouri, Dataset for file fragment classification of video file formats, BMC Res Notes 13 (2020) 213, https://doi.org/10.1186/s13104-020-05037-x.

[26] S. Garfinkel, P. Farrell, V. Roussev, G. Dinolt, Bringing science to digital forensics with standardized forensic corpora, Digit Invest 6 (2009), https://doi.org/10.1016/j.diin.2009.06.016. S2−S11.

[27] Aaron, O.S. Sitompul, R.F. Rahmat, Distributed autonomous Neuro-Gen Learning Engine for content-based document file type identification, in: 2014 International conference on cyber and IT service management (CITSM), 2014, pp. 63−68, https://doi.org/10.1109/CITSM.2014.7042177.

[28] T. Xu, M. Xu, Y. Ren, J. Xu, H. Zhang, N. Zheng, A file fragment classification method based on grayscale image, 2014, https://doi.org/10.4304/jcp.9.8.1863-1870.

[29] P. Penrose, R. Macfarlane, W.J. Buchanan, Approaches to the classification of high entropy file fragments, Digit Invest 10 (2013) 372−384, https://doi.org/10.1016/j.diin.2013.08.004.

[30] Q. Chen, Q. Liao, Z.L. Jiang, J. Fang, S. Yiu, G. Xi, et al., File fragment classification using grayscale image conversion and deep learning in digital forensics, IEEE Security and Privacy Workshops (SPW), 2018, pp. 140−147, https://doi.org/10.1109/SPW.2018.00029.

[31] N. Beebe, L. Liu, M. Sun, Data type classification: hierarchical class-to-type modeling, in: Advances in digital forensics XII, Springer International Publishing, 2016, pp. 325−343, https://doi.org/10.1007/978-3-319-46279-0_17.

[32] M. Portaz, J. Poignant, B. Mateusz, Construction et évaluation d'un corpus pour la recherche d'instances d'images muséales, En Recherche D', 2017. https://hal.archives-ouvertes.fr/hal-01802259/.

[33] O.O. Aremu, D. Hyland-Wood, P.R. McAree, A machine learning approach to circumventing the curse of dimensionality in discontinuous time series machine data, Reliab Eng Syst Saf 195 (2020) 106706, https://doi.org/10.1016/j.ress.2019.106706.

[34] C.E. Shannon, A note on the concept of entropy, Bell System Tech J 27 (1948) 379−423.

[35] C.E. Shannon, A mathematical theory of communication, Bell Syst Tech J 27 (1948) 623−656, https://doi.org/10.1002/J.1538-7305.1948.TB00917.X.

[36] T. Shang, R. Liu, C. Fang, J. Liu, Quantum network coding based on entanglement distribution, in: Artificial intelligence and security, Springer International Publishing, 2019, pp. 13−24, https://doi.org/10.1007/978-3-030-24268-8_2.

[37] C. Grajeda, F. Breitinger, I. Baggili, Availability of datasets for digital forensics − and what is missing, Digit Invest 22 (2017), https://doi.org/10.1016/j.diin.2017.06.004. S94−S105.

[38] I. Taleb, M.A. Serhani, R. Dssouli, Big data quality: a survey, in: 2018 IEEE international congress on big data, BigData Congress), 2018, pp. 166−173, https://doi.org/10.1109/BigDataCongress.2018.00029.

[39] A. Zheng, A. Casari, Feature engineering for machine learning, principles and techniques for data scientist, 2018.

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al., Scikit-learn: machine learning in Python, J Mach Learn Res 12 (2011) 2825−2830. http://jmlr.org/papers/v12/pedregosa11a.html.