Classifying the classifiers for file fragment classification

Ashim Shahi

Master's Thesis

22nd August 2012

Master Software Engineering

Universiteit van Amsterdam

Thesis Supervisor: Jeroen van den Bos

Internship Supervisor: Jurgen Vinju

Institute: Centrum voor Wiskunde en Informatica

Universiteit van Amsterdam

CWI

Contents

Tables in the document

Abstract

There has been a lot of research in the field of classification for file types. Researchers have attempted to classify full files for use in virus scanners, fragments of files to identify the type of data contained within the files. The research so far however seem to lacking at the amount of data that they have used in their analysis. In this thesis, we aim to look at how well the different algorithms that have been developed do in the case when we are looking at large amounts of data. In order to be practical, we have chosen to use around 100GB of data, which we consider to be large enough especially in light of the largest data set being used for far being 750MB.

We have used 4 different algorithms and 10 file types for the purposes of our experiment. We have sampled and tested the data using 4KB fragments. The importance of this size lies in the fact that most new storage media use it as their sector size and fragments of files are stored one sector at a time. We aim to find if there is an algorithm that could be called best in terms of accuracy and performance. We also try to find the amount of data that each algorithm needs to provide good results, since if new file types are created, we would not have enough data at the start for classification if we require large amount of data for good results.

The results of our experiment show that there are clearly some algorithm that have better performance, but when adding accuracy into the equation, there are only a few data types that are actually benefitted by the classification. There is also no definite answer on how much data is needed to train those algorithms since we do not see any consistent pattern that allows us to believe that adding more sample data would not affect the classification results. This has led us to question whether it is even possible to get fast and cheap classification for file fragments.

Acknowledgements

I would like to thank Jeroen van den Bos for his constant feedback, supervisor, continued support through-out the writing of this thesis. I would like to thank Jurgen Vinju for offering me the opportunity to work as an intern at CWI for the completion of my thesis. I would also like to thank Paul Klint, Bas van Vlijmen and Tijs van der Storm for the valuable feedback that they provided during the conceptualization of this thesis.

Furthermore, I am indebted to all the members of SWAT team at the CWI and the members of Master Software Engineering program at Universiteit van Amsterdam for showing me around the wonderful world of Software Engineering. It has been a wonderful learning experience.

1. Introduction

1.1. File Carving

An important part of Digital Forensic analysis is recovery of files from a storage media. In cases where the storage media are undamaged and have their file system metadata intact, this process is rather simple since the data can be recovered by parsing the file system metadata. If however, the file system metadata are corrupt, the only way to get meaningful data out of the storage media would be reconstructing the files based on their raw content. This process is referred to as File Carving. The advantage of file carving from digital forensic perspective actually lies in the fact that it can be used to reconstruct deleted files or file that have been tampered with intentionally, as in case of files that are hidden or obfuscated, so that the operating system and/or the file system are unable to reconstruct them.

1.2. File Fragments

In the absence of file system metadata, the raw contents of a storage media are no more than fragments of files. We say fragments because, the design of storage media do not accommodate files being stored as a single unit, unless they smaller in size compared to the sector size of the media. In the best case scenario, the fragments are in contiguous sectors in the storage device. In the worst case scenario, we have no idea where the fragments are which is the case with most of the flash drives and solid state disk available today. A likely scenario is presented in the picture below. In these media, the writing of contents are spread uniformly across all chips to maximize the life of the disk. In these cases there is no way to know where a file begins and where it ends. This is the inherent problem with file carving. As an example we can see from the figure below how files make be fragmented. The numbers indicate different file types which the columns indicate sector size boundaries.

1	2	5	4	2	1	3	1	1	1

The size of file fragments are determined by the sector sizes of the media devices. The most commonly found sector sizes in the storage medias still in use today are 512 bytes, which is used in mostly older media devices and 4096 bytes, which is the default size in most of the current media devices. Another important fragment size would be 1460 bytes which is the size of a network packet, which like its media device counterpart are also one of the of interest for digital forensic analysts and suffer from the same problems in case the information containing the type of the data in the packet is lost. The importance of these fragment sizes are highlighted by their inclusion in the 2012 Digital Forensic Research Workshop Challenge.

1.3. Fragment classification

A possible solution to the problem of file carving would be the classification of the fragments. Fragment classification is the process of separating unknown file fragments into known file types based on the raw content that is observed in the fragment. Classification of the fragments provides the advantage of reduction in the number of fragments that need to be considered for reconstructing files thus increasing the performance of the already expensive file carving process [1]. Classification of fragments is an ongoing research area and many algorithms have been developed for fragment classification with mixed results. The algorithms range from statistical measurements to machine learning.

The research so far however suffers from some problems. The data set employed by the researchers has been rather small with cases where researchers have used around 100 files per file type while other have used 750MB of data. The results of the experiments are not actually comparable because of difference in the data sets used, the different types of files used, the difference in the method of differentiating between sample set and test set to name a few. We will discuss more on this in the next sections.

From a software engineering point of view we would like to know how these algorithms perform when they are compared keeping all other variables the same. We would like to know which algorithm, if any, would be best for use depending on the file type we are interested in, or depending on our performance and accuracy requirements.

1.4.Research question

The work in this thesis will revolve around the following research question:

- Which algorithm is the best in terms of performance and accuracy?

Other questions we attempt to answer in this thesis:

- Is it possible to train some algorithms quicker than others with less data?
- Can we find a way to ensure that the sample data we have collected is a representative set for the file type that is being analyzed?

2. Related Work

McDaniel and Heydari [3] came up with the Byte Frequency Analysis (BFA) and Byte Frequency Cross-Correlation (BFC) algorithms to identify file fragments. Their approach in BFA was to create a histogram of the frequency of ASCII codes (0..255) for each file that needed to be classified. Based on the histogram, 256-element vectors were created for each file types which were clustered and then used to identify unknown file types. BFA only achieved a true positive detection rate of 27.50% for whole files which is not a good rate for practical purposes. The BFC algorithm, like BFA, created histogram too but it also analyzed the average difference in frequency between all byte pair to identify characters that occurred with nearly identical frequencies like "<" and ">" in HTML files, which are used as a pair to denote HTML tags. The true positive detection rate for BFC algorithm was found to be around 46% which is still considered to be inadequate for practical purposes.

Karresand and Shahmehri [4] developed the Rate of Change (RoC) metric, defined as the difference of the ASCII values of consecutive bytes. The focus of the research was to increase the accuracy of recognition of jpeg files. JPEG format uses 0xFF as an escape character for metadata tags and the encoder inserts an extra 0x00 after every 0xFF byte in a jpeg file. Roussev and Garfinkel [2]note "This produces a very regular, unique and easily exploitable pattern - 0xFF00 - which has a very high ROC". The true positive rates of RoC for other file types aren't nearly as good as for the jpeg files with false positive rates exceeding the true positive rate for some file types.

The work of Erbacher and Moody [5] try to classify formats through a statistical perspective using standard statistical measurements like averages, distributions and higher momentum statistical measurements. Their work shows that these approaches can be used to classify broad classes of files but are unable to classify files of the same class with little differences like text files against html files. However, the focus of their research was in identifying the primitive data types contained within files, rather than classifying the file or fragments of the file themselves.

Veenman [6]using BFD (Byte Frequency Distribution) approach along with Shannon entropy and Kolmogorov complexity was able to get very high detection rates for html and jpg files but the results for zip and executable files weren't as good.

Li et al [7] have used the n-gram analysis method where they measure the byte frequencies to create a fingerprint based on a centroid containing the mean and standard deviation of the byte frequencies. The classification is then done based on Mahalanobis distance function. They perform their research on the first 20, 200, 500 , 1000 bytes of each file type and also on the full files. They report a true positive detection rate of over 99% percent when using the first 20 bytes which is explained by the fact that these are mostly header data for the files and would be consistent across files. They report success rates of around 60-90% percent when using full files.

Conti et al [8] have used the Shannon Entropy, Mean, Chi Square and Hamming Weight to try to find the contents of a binary document. They look at these four vectors and try to classify the contents of the files into one of many classes like compressed-text, compressed image, random, text, bitmap etc. They report successful detection rates of over 82% for all their classes.

Roussev and Garfinkel [2] discussing the limitations of statistical methods and propose the need for specialized approaches for each type of file. Through their examination of zlib, jpeg and mp3 files the conclude that machine learning and statistical methods are less likely to be successful due to their dependence on easily detectable patterns. They also point out that current methods do not use enough sample data and do not distinguish between primitive types and compound data formats.

3. Experiment

As mentioned in the previous section, a number of different algorithms have been developed over the years to address the problem of identifying file/fragment types. The different algorithms have reported varied successful classification rates for different types, but we do not know how well these algorithms compare against each other for the following reasons:

- The data used for sampling and testing are different. The effect of having different data sets leads us to question whether the presented results would hold while using another data set. The differences between classification rates for the different algorithms also cannot be put into proper perspective without the same data set.
- The amount of data used for training and testing the algorithms seem too small in our opinion. With the decrease in hardware costs and the increase in the size of media storages available today, we believe that the amount of data used by the algorithms so far do not form a representative set for the data types. Sample sizes are an important aspect in classification since being able to train with small sample sizes means that new file types could be added to classification, which having a large set would allow us to cover the different cases where the file types are used in different ways (like pdf file having text only, both text and image or image only) that exist within common file types.

In the following sections we discuss how we have structured our experiment. Section 3.1 discusses the collection of data for the experiment, section 3.2 discusses our choice of the file types, section 3.3 discusses the algorithms we have chosen for our experiment and the difference between them that makes them difficult to compare against each other and finally 3.4 discusses our experimental setup.

3.1.Data Collection

Getting hold of data that can be used as a representative set for digital forensic analysis has never been an easy task. As can be seen in the algorithms mentioned in the previous section, most of the research focuses on trying to gather data from popular search engines using wildcards or researchers use data that they have themselves created. Just using the data downloaded randomly from the internet is also not feasible since there are many copyright laws and licenses protecting the rights of the owners of the data. To address the issue, Garfinkel et al [7] have created a digital forensic corpora available at Digital Corpora [8].

We have used Digital Corpora as our primary data source. We downloaded 1000 zip files, each with 1000 files each, giving up a total of 1 million files with a size of around 310 GB. The fact that the corpora was collected by researchers in the field of digital forensic leads us to believe that the file types present are relevant to digital forensics. The availability of the data us makes sure that work done in this thesis can be reproduced and tested with the same data set.

For two of the data types, namely PNG and OGG, the data were collected from Wikipedia by Jeroen van den Bos and Tijs van der Storm [9].

For MP4 data type, we have collected data from Academic Earth [10]. The website contains videos of lectures from universities that provide open video courses for students. We manually looked at the licenses for the lectures and chose those whose license included freedom to copy the files for non-commercial purposes. We then downloaded those files for analysis.

3.2.Type selection

Looking at the data, it was visible that not all file types were represented equally. There were some types which consisted of over 120GB of data (after extracting the zip files) while other with less than 1GB. Table 1 lists the file types that have at least a few gigabytes of data from the Digital Corpora set.

File Type	Size (in GB) (After extracting the data)
xml	7.76
pdf	124.6
ps	26.6
csv	3.26
doc	19.2
gif	2.37
log	4.11
pps	3.54
xls	18.3
gz	8.46
text	49.3
zip	311*
html	10.9
ppt	120
jpg	40

Table 1: Some file types from Digital Corpora

File Type	Sample set		Test set	
	Size (in GB)	Number of files	Size (in MB)	Number of files
jpg	9.01	28,264	925	1,995
pdf	9.01	17,976	924	1,859
png	9.02	66,381	925	4,949
ogg	9.02	6,786	925	566
doc	9.01	17,891	923	1,953
mp4	9.00	72	875	2
xls	9.02	12,349	924	453
text	9.02	12,465	925	1,358
zip	8.99	32	920	3
ppt	9.02	5,020	922	227

Table 2: Chosen file types and sizes divided by usage

* The size of the zip files actually correspond to the original zips that were downloaded. We have used the original zip files as the data for the zip files analysis.

In choosing the file types for this thesis, we had two simple considerations. The first was to make sure that we had the same amount of data for each of the file type chosen. The second was to use as much data as we could possibly find. The minimum amount of data that we found allowing us to keep ten different file types was 18 GB per file type giving us a total of around 180 GB of data to analyzed. Initially, we divided the 18 GB per file type into two equal sets of 9GB, one to use as the sample set and other to use as the test set. We later had to decrease the size of the test set to 0.9 GB per file type due to performance issues with running the experimental setup that will be explained in detail in section 3.4. In total, we are using around 90 GB of data as sample data and 9 GB as the test data.

The decision to choose these file types were also in part due to their inclusion in the Digital Forensic Research Workshop Challenge 2012. The only exceptions in the list are ogg and mp4 which we chose as representatives of audio and video types respectively.

The chosen file types along with their sized and number of files used for both sample set and the test set are listed in Table 2.

3.3.Chosen algorithms and their differences

Out of the algorithms mentioned in section 2, we have chosen the following four algorithms for comparison in this thesis:

- Byte Frequency Analysis
- Rate of Change
- n-gram Analysis
- The algorithm of Conti et al

The motivation for choosing the above mentioned algorithms lies in the facts that they are all statistical analysis methods and they are all relatively less complex than the other algorithms. We say less complex in the sense that they all require less computation time. Computation time is an important factor in the considerations of the algorithms since there are situations where a digital forensic analyst would compromise some accuracy in order to get a general idea of the contents in a shorter time frame and also because file carving is already a very expensive process.

Table 3 lists some of the differences that we have observed that make the comparison of the 4 chosen algorithms very hard. The algorithms employ different tactics to build up the fingerprints for each file types like byte frequency algorithm and n-gram analysis use full files in order to build their fingerprints whereas Rate of Change uses 4KB fragments and the algorithm of Conti et al uses 1KB fragments. The difference could be explained by the fact that byte frequency methods and n-gram analysis were meant to work for file scanning like in a virus detector or network packet analyzer whereas the algorithm of Conti et al was primarily created to find data types within a single file. That also explains why the test set used by Conti et al only consists of one Microsoft Word document file and a Mozilla browser dump.

Another interesting difference between the algorithms is that in byte frequency algorithm, the frequencies of byte values are normalized first by the maximum occurring byte value and second with a companding function which places emphasis on the byte values that have very low frequencies too. Li et al mention that they think this method of normalization in byte frequency analysis where size of the files are not considered may not be suitable and propose normalizing by the number of bytes that are analyzed but do not provide any suggestion to whether they did any such normalization in their work or not.

Rate of Change differs from the other algorithms in a number of different ways. Firstly, it is the only algorithm in the list where the authors have chosen to pad fragments less than 4KB with 0 values. Secondly, the research from the authors seem to focus on finding JPEG file types, which is evidenced by their use of a JPEG extension. The published results also do not provide us with any concrete idea of how the algorithm fared in terms of other types since they only present graphs with True Positive rate vs False positive rate. The authors also mention that they experimented with different threshold values but do not provide us with the threshold values that provided them with the best result except for their run with the JPEG extension included.

Algorithm	Author(s)	Measures (for each fingerprint)	Measures (between fingerprints)	Comparison method	Sample set	Execution of sample set	Training set	Execution of test set	Further notes
Byte Frequency Analysis	McDaniel, M. and Heydari, M.H	Frequency of byte values	Mean of the byte values and correlation between each byte frequencies	Generate score based on the difference between the frequency of byte values	30 file types but no mention of the number of files	Full files	30 file types, 4 files per file type	Full files	
Rate of Change	Karresand, M. and Shahmehri, N.	Frequency of the difference between consecutive bytes	Mean and standard deviation for each byte	Quadratic distance measure	Not mentioned	4KB fragments	57 files totaling 72.2 MB (all files concatenated into 1 file), fragments with less than 4KB padded with 0 values	4KB fragments	Results presented were with a JPEG extension. Threshold values not present except for the JPEG extension run.
n-gram Analysis	Li, W. J. et al	Frequency of byte values	Mean and standard deviation	Mahalanobis distance function between mean and standard deviation	8 file types, 100 files per file type	Truncation to different sizes (first 20, 200, 500, 1000 bytes) and Full files	8 file types with 100 files per file type	Truncation to different sizes (first 20, 200, 500, 1000 bytes) and Full files	Do not mention if they normalize or not.
Algorithm of Conti et al	Conti, G. et al	Shannon Entropy, Arithmetic mean, Chi square and Hamming weight	Mean of the 4 vectors	Euclidean distance and k-nearest neighbor	14000 fragments	1KB fragments, from the middle of the file	10.3 MB Microsoft Word 2003 document and 41.4MB Firefox browser dump	1KB fragments, sliding window	Developed for identifying content types within files

Table 3: Differences observed in the 4 chosen algorithms

3.4. Experiment Setup

Based on the differences that we have noted in the previous section, we decided on the following settings for the experiments in this thesis:

- Fingerprints would be created and tested on 4KB fragments. As mentioned in section 1, 4KB is the sector size of most storage media today, so we think creating fingerprints and test based on this size would be the best.
- The sample set and the test set would contain different files.
- Due to the lack of the best threshold values in case of Rate of Change, we decided to classify a fragment to the type which had the nearest distance to a fingerprint.
- For classification with the algorithm of Conti et al, we decided to only use the Euclidean distance and neglect the k-nearest neighbor classification. The decision was made in accordance with the fact that we want to find fragment types in storage media that may not store fragments continuously like Hard disk drives do.

The experimental setup is focused on measuring three things:

- How do the algorithms compare to each other in terms of accuracy and performance? This is the main question and the other measurements will also focus on computing these requirements.
- How much sample data is needed for a good classification for each algorithm?
- Can be combine two or more algorithms to give us an increased accuracy without losing too much on performance?

To measure these things we have divided our experimental setup into 4 steps.

3.4.1. Creating fingerprint from the sample

Creating fingerprint for each of the instance is the first thing before the experiment can actually start running. Since, we are using 4 algorithms and 4 instance per algorithm for 10 different file types, we get a total of 160 fingerprints.

3.4.2. Amount of data needed

For the chosen algorithms, we want to know if one particular algorithm can be trained with less sample data while achieving similar if not better results for classification. This experiment will also allow us to see whether adding more data to the fingerprint will improve the classification, which is what we would expect. To perform this experiment, we randomly divide the sample data into 10 parts. We use 4 different instances for each algorithm, containing 10% (0.9GB), 20% (1.8GB), 50% (4.5GB) and 100% (9GB) of the sample data in order to generate the fingerprints for each file type. With each increasing sample size, we make sure to keep the previous sample size intact, i.e. when we sample using 20% of the data, we include the 10% sample in this 20% of the data. Each sample size will be a different instance for testing and will provide us with a single value for the classification. Since, we do this for 4 different algorithms, it provides us with a 16 instances for comparison between how well the algorithms perform against each other for different sample sizes and also how well each algorithm compares to itself with different sample sizes.

3.4.3. Similarities between the classifications

We use the Jaccard Index to find the similarities between the classifications from each classification results. The primary objective of calculating the Jaccard Index is to see if the classification of the different algorithms have less intersection which would allow us to combine the algorithms for better accuracy. Since, we also measure the performance of each algorithm, we can see if the better accuracy is acceptable in terms of performance when combining the algorithms. The Jaccard Index, will be used to measure the number of fragments that were classified to be of the same type irrespective of whether the classification was correct or not, though the correct classification ones would be the ones to focus on. Since we have 16 results for comparison, we will have a 16x16 matrix of values to compare the similarity between each instance to the others. We do expect, the different instances of the same algorithm to provide similar if not exactly the same index.

The results could point out specific cases of fragments that are confused by 1 or all algorithms, which would mean these methods would not be useful for those fragments, they would need special attention.

3.4.4. Measuring performance and accuracy

In order to make the calculation for the Jaccard Index without any doubt about that the same fragments have indeed been compared, we need to make sure that all the instances run on each fragment before a new fragment is considered. To achieve this effect, the tool has been designed in a way so that 1 fragment is read, the data needed for each algorithm is computed, classification is done for each of the instances, Jaccard Index is calculated. All the previously mentioned steps are timed, once for the read operation, separately for the other steps per instance and then added up to calculate the final time for processing the fragment. The Jaccard Index calculation times are not added since they are not part of the original algorithms. The time for processing each fragment will act as the performance measure for each instance.

For accuracy, we create an 11x11 matrix, one for each file types and an extra one for the “Unclassified” case. The result of the classification is compared to the file type of the fragment which will be known and a confusion matrix created bases on 11x11 matrix. The 11th column though will not have any data as we do not test for any data that we do not know the type for.

4. Results

In this section we only present the results we obtained from our experiment, the analysis of the results will be presented in the next section. Section 4.1 discusses the classification results of the different instances i.e. accuracy, section 4.2 discusses the similarities between the classification results looking at the Jaccard Index and finally section 4.3 discusses the performance of the algorithms.

4.1. Accuracy

The accuracy of the classification of the different algorithms are presented below as a percentage confusion matrix measured to within 1 decimal place for readability (tables with the number of fragments for each algorithm, on which the detection rates are based, can be found in Appendix A). The rows represented the file type that a fragment was classified as, while the columns represent the actual types of the fragments. The shaded regions are the True Positives detected. The numbers indicate the ratio of classified fragments to total fragments for each file type. Within each cell, the 4 comma separated values identify the classification rates for the 4 different sample sizes that were used for the algorithm. For example, a value of 15, 20, 15, 20 represents a classification rate of 15% with 10% sample data used for fingerprint, 20% with 20%, 15% with 50% and 20% with 100%.

In the following paragraphs, we highlight some of the most noticeable results from Table 4, 5, 6 and 7. We will follow up with a detailed analysis of the results in section 5.

As can be seen from Table 4, Byte frequency algorithm with only 10% of sample data performs very well for text and zip types while it performs extremely poorly for ppt and pdf types. With an increase in the sample size, byte frequency algorithm shows a slight increase in text and zip types, a significant increase in the detection of jpg types and a significant decrease in the detection of xls types. The increase of sample size has no effect on pdf and ppt types. Further increase in the sample size increases the detection rate of xls types significantly. Adding more sample data once again decreases the detection rate of jpg while the detection rate of xls types increases by more than double.

From Table 5 we see the rate of change algorithm with 10% of the total sample data shows good detection rate for ppt and xls types. An interesting observation here is that rate of change classifies most of the fragment types as ppt type. The increase in sample size doesn't show much effect for the ppt and xls types, while we see an increase in the detection rate of text file and a decrease for pdf and doc types. Upon further addition of sample data, the detection rate for ppt and pdf types decrease a little while the detection rate for text increases slightly. The only significant change with adding more data to the rate of change algorithm seems to be high increase in the detection rate of text files.

Table 6 tells us that the n-Gram analysis shows extremely high detection rate for text and xls types and a high detection rate for ppt types when using a sample size of 10%. Similar to rate of change algorithm however it also seems to confuse most of the types with the ppt type. On increasing the sample size to 20%, we see a decrease in detection rate of ppt types. Further addition to the sample size shows a significant decrease in the detection of ppt types while maintaining the high detection rates for xls and text types. Similarly, increasing the sample size to 100% decreases the detection rate of ppt types while keeping the good detection rate for text and xls.

Table 7 shows the algorithm for Conti et al has good detection rate for mp4 types and average detection rate for text and zip types when using 10% of the total sample data. Adding more sample data, we see an extreme drop in the detection rate of mp4 types. The detection rate of ogg types seems to almost double. Using 50% of the sample data, the detection rate for mp4 types has gone down to 0, while the detection rate for text and ogg types have increased considerably. Using 100% of the sample data, the detection rate for text type falls slightly while the detection rate for ogg types increases further.

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
zip	33.1, 34.6, 34.8, 36	86, 88, 88.2, 89.5	1.9, 2.1, 2.1, 2.2	17.9, 18.9, 19.1, 20	22, 26.1, 26.2, 28	0, 0, 0, 0	48.1, 50.9, 51.3, 53.7	33.5, 34.9, 35.9, 42.1	6.7, 8.7, 8.8, 9.8	62.8, 69.2, 69.5, 72.1
text	15.7, 17.1, 19.7, 16.9	0.1, 0.2, 0.3, 0.1	96.2, 96.3, 96.3, 96.2	47.7, 57.6, 55.8, 42.7	4.7, 4.6, 5.9, 4.3	43, 79.9, 67.8, 23.6	5.5, 9.4, 10, 5.8	1.1, 1.2, 4, 1.7	10.4, 14.4, 20.4, 17.9	2.3, 2.5, 3.1, 2.5
doc	2.1, 0.5, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0.5, 0.2, 0, 0	0.6, 0, 0, 0	0, 0, 0, 0	0.4, 0.1, 0, 0	0.1, 0, 0, 0	8.2, 3.6, 0, 0	0.3, 0.1, 0, 0
mp4	10.1, 8.9, 10.1, 11.2	4.5, 2.9, 2.9, 3.2	0.4, 0.3, 0.3, 0.3	4.1, 3.3, 3.8, 4.3	27.2, 23.4, 24.9, 27	0, 0, 0, 0	12.3, 9.2, 9.4, 10.3	25.2, 13.4, 13.6, 15.1	18.2, 18.3, 20.8, 22.4	11.4, 7, 6.8, 7.3
xls	11.4, 11.5, 9.3, 11.5	0.3, 0.3, 0.2, 0.3	0.3, 0.2, 0.2, 0.3	17.9, 8.3, 10.4, 23.3	0.2, 0.2, 0.1, 0.1	56.8, 19.9, 32.1, 76.2	10.9, 7.3, 7.1, 10.9	4.4, 4.6, 2.3, 4	6.4, 6.7, 4, 4.1	1.8, 1.7, 1.3, 1.7
ppt	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
jpg	2.6, 6.5, 7.6, 6.3	1.3, 3.9, 4.5, 3.2	0.2, 0.4, 0.5, 0.4	2, 4.2, 4.6, 3.6	0.2, 5.8, 9.4, 8.8	0, 0, 0, 0	4.6, 10.1, 11.6, 9	9.7, 30.4, 31.8, 25.1	3.4, 6.2, 9, 8.2	1.9, 7.7, 9.6, 7.1
ogg	20.6, 18.8, 16.8, 16.1	3, 2.5, 2, 1.7	0.2, 0.2, 0.2, 0.1	6.5, 5.8, 4.9, 4.8	39.7, 35.4, 29.5, 27.9	0, 0, 0, 0	10.9, 9.4, 7.7, 7.3	16.3, 13.8, 11, 10	40.2, 36.8, 32.8, 33.3	6.4, 5.5, 4.4, 4.1
png	4.1, 1.7, 1.4, 1.5	4.5, 1.9, 1.6, 1.6	0.4, 0.1, 0.1, 0.1	2.8, 1.1, 0.9, 0.9	5, 4.1, 3.8, 3.5	0, 0, 0, 0	6.8, 3.2, 2.6, 2.7	9.4, 1.3, 1.1, 1.7	6.2, 5, 3.8, 4	12.8, 5.9, 4.9, 4.8
UnClassified	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0.1, 0.1, 0.1, 0.1	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0

Table 4: Classification results of Byte Frequency Analysis

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	24.1, 22.9, 21.9, 21.6	0.7, 0.2, 0, 0	0, 0, 0, 0	0.8, 0.6, 0.4, 0.4	0.3, 0, 0, 0	0, 0, 0, 0	1.4, 0.9, 0.5, 0.5	1.8, 0.8, 0.3, 0.3	0.3, 0.1, 0, 0	1.3, 0.6, 0.3, 0.3
zip	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
text	3.9, 2.9, 1.8, 2.2	0, 0, 0, 0	64.7, 72.6, 74.9, 83.9	7.9, 6, 8.3, 8.8	0, 0, 0, 0	1, 0.8, 0.9, 1	0.1, 0.1, 0.1, 0.1	0, 0, 0, 0	0, 0, 0, 0	0.4, 0.4, 0.4, 0.4
doc	2.5, 2.2, 2.9, 2.9	0.1, 0.1, 0.2, 0.2	12.9, 1.3, 1.4, 1.5	44, 39.1, 39.3, 40.9	0.1, 0, 0, 0	3.4, 2.3, 1.7, 2.1	8.4, 7.2, 8.3, 8.6	1.3, 1.4, 1.8, 1.6	0.2, 0.3, 0.5, 0.5	2.9, 3.4, 4.5, 4.3
mp4	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
xls	10, 12.3, 13.2, 12.6	0, 0, 0.1, 0.1	18.7, 22.4, 20, 11	9.4, 17.9, 16.5, 14.2	0.1, 0.2, 0.2, 0.2	94.8, 96.3, 96.8, 96.7	3.1, 5.5, 5.9, 5.5	0.4, 0.6, 0.8, 0.8	0, 0, 0.1, 0.1	0.3, 0.5, 0.8, 0.8
ppt	59.2, 59.5, 60, 60.4	99, 99.4, 99.4, 99.	3.4, 3.4, 3.4, 3.4	37.7, 36.1, 35.3, 35.4	99.3, 99.6, 99.6, 99.6	0.6, 0.3, 0.4, 0	86.7, 86.2, 84.9, 85.1	96.2, 97, 96.9, 97.1	99.3, 99.4, 99.2, 99.3	94.7, 94.8, 93.7, 94
jpg	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
ogg	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
png	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
UnClassified	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0

Table 5: Classification results of Rate of Change

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	9.3, 10, 7, 5.4	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
zip	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
text	8.3, 9.3, 9.1, 9.4	0, 0, 0, 0	94.3, 95.2, 95, 95.3	7.2, 7.5, 7.4, 8	0, 0, 0, 0	0.3, 0.5, 0.4, 0.6	0.2, 0.2, 0.2, 0.2	0.1, 0.1, 0.1, 0.1	0, 0, 0, 0	0.2, 0.2, 0.2, 0.2
doc	17.6, 20.3, 27.9, 33.6	1.9, 2.9, 4.8, 5.8	0.3, 0.4, 0.4, 0.7	10, 10.9, 13.1, 16.3	6.9, 19.2, 46, 54.8	0.1, 0.1, 0, 0	9.9, 12.3, 20.2, 24.5	9.7, 12.3, 25.3, 33.4	8.1, 11.3, 22.3, 28.6	3.1, 4, 9, 13.1
mp4	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
xls	6.7, 5.6, 7.1, 5.8	0, 0, 0, 0	1.9, 0.9, 1.3, 0.8	48.2, 48.3, 50.7, 48.9	0.3, 0.3, 0.3, 0.3	99.3, 99.2, 99.4, 99.2	8.3, 8.3, 9, 8.6	0.4, 0.4, 0.6, 0.5	0.1, 0.1, 0.4, 0.2	1.6, 1.6, 1.9, 1.8
ppt	57.8, 54.5, 48.7, 45.6	98, 96.9, 95, 94.1	3.4, 3.3, 3.1, 2.9	34.4, 33.1, 28.6, 26.6	92.7, 80.3, 53.6, 44.8	0, 0, 0, 0	81.5, 79, 70.5, 66.6	89.5, 87, 73.7, 65.8	91.6, 88.5, 77.2, 71	95, 93.9, 88.7, 84.6
jpg	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
ogg	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
png	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0
UnClassified	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0

Table 6: Classification results of n-Gram analysis

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	5.8, 4.2, 0.1, 4.1	0.5, 0.3, 0, 0.1	0.4, 0.3, 0, 0.2	2, 1.4, 0.1, 1.8	0, 0, 0, 0	0, 0, 0, 0	2.9, 2.1, 0.3, 2.1	1.2, 0.9, 0.3, 0.7	0.7, 0.5, 0, 0.2	1.2, 0.8, 0.1, 0.9
zip	15.1, 15.4, 15.5, 15.3	44.1, 45.2, 45.5, 44.9	0.1, 0.1, 0.1, 0.1	4.1, 4.2, 4.3, 4.2	3.6, 3.8, 3.8, 3.7	0, 0, 0, 0	13, 13.5, 13.6, 13.3	0.1, 0.1, 0.1, 0.1	0.8, 0.9, 0.9, 0.9	10.4, 10.7, 10.9, 10.6
text	6.1, 6.2, 8.9, 8.5	0, 0, 0, 0	57.8, 59.5, 82.3, 80.4	41.3, 40.9, 41.8, 40.7	0.2, 0.2, 0.2, 0.2	50.9, 49.2, 49.2, 47.3	3.6, 3.5, 3.5, 3.3	0.3, 0.3, 0.4, 0.4	0, 0, 0, 0	1, 1, 1.2, 1.1
doc	3.5, 3.8, 4.3, 4.2	0, 0, 0.1, 0.1	11.5, 10.5, 6.5, 8.4	11.8, 11.7, 8.4, 10.5	0, 0, 0, 0	15.4, 13.8, 2.8, 7.1	1.9, 2, 2.3, 2.3	0.4, 0.5, 0.6, 0.6	0.1, 0.1, 0.1, 0.1	0.5, 0.6, 0.8, 0.7
mp4	22.4, 16.7, 7.7, 9	27.3, 16.2, 1.2, 1.3	1.6, 0.9, 0.1, 0.1	11.6, 6.2, 1.3, 1.6	82.4, 7.1, 0, 0	0, 0, 0, 0	32.3, 17.1, 2.1, 2.7	61.5, 28.2, 1.2, 1.5	49.5, 25.9, 2.5, 4.6	44.8, 27.9, 4.1, 4.7
xls	3.1, 3.1, 1.1, 1.3	0, 0, 0, 0	26.5, 25.7, 7, 7	7.1, 7.9, 10.9, 9.7	0, 0, 0, 0	33.4, 36.7, 47.7, 45.3	4.4, 4.7, 5.1, 5.2	0, 0, 0, 0	0, 0, 0, 0	0.2, 0.2, 0.1, 0.1
ppt	6.4, 6.9, 10.1, 7.4	0.3, 0.3, 0.6, 0.6	0.3, 0.3, 0.6, 0.4	2.9, 3.1, 3.8, 2.5	0, 0, 0, 0	0, 0, 0, 0	3.3, 3.5, 4.6, 3.4	1, 1.1, 1.5, 1.3	0.6, 0.8, 1.3, 1.3	1.4, 1.5, 2.1, 1.5
jpg	18.6, 20, 19.8, 13.6	20.1, 30.1, 29.2, 13.2	1.3, 1.7, 1.7, 1	14, 15.9, 15.8, 10.9	3.1, 5, 5.4, 2.3	0, 0, 0, 0	29.6, 35.1, 34.9, 23.2	32, 35.5, 35.6, 24.2	29.1, 36, 37.2, 25.2	31.7, 42.6, 42.5, 23.6
ogg	11.8, 13, 22.9, 28.8	0.8, 5.8, 21.9, 38.3	0.1, 0.3, 1.2, 1.9	1.9, 4.4, 9.5, 14.4	9, 83.2, 90, 93.1	0, 0, 0, 0	3.2, 12.4, 27.8, 39.6	1.8, 27.5, 54.4, 65.8	18.2, 34.5, 56.2, 66.4	3.3, 9.6, 34.1, 53
png	6.8, 10.1, 9.1, 7.4	6.5, 1.6, 1, 1	0.1, 0.1, 0.1, 0.1	2.7, 3.8, 3.6, 3.1	1.2, 0.3, 0.2, 0.2	0, 0, 0, 0	5.3, 5.7, 5.2, 4.4	1.1, 5.4, 5.4, 4.8	0.6, 1, 1.3, 1	5, 4.4, 3.7, 3.1
UnClassified	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0.1, 0.1, 0.1, 0.1	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0

Table 7: Classification results of algorithm of Conti et al

4.2. Similarity between the algorithms

This section presents the results of the Jaccard Index calculated for a selected few instances against each other. The numbers presented are Jaccard Indices for the two algorithms presented in a percentage form. We chose to present the comparison results of the 100% sample data sizes as representing the similarity of the classification results for each algorithm. The rows represent the classified types and the columns represent the actual types of the fragments.

Table 8 provides the Jaccard Indices of byte frequency algorithm with 100% sample size when compared with rate of change 100% sample size, n-gram analysis 100% sample size and the algorithm of Conti et al 100% sample sizes. The comma separated values in the cells represent the above mentioned instances in order.

Table 9 provides the Jaccard Indices of rate of change algorithm with 100% sample size when compared to byte frequency algorithm 100% sample size, n-gram analysis 100% sample size and algorithm of Conti et al 100% sample size represented by the comma separated cells respectively.

Table 10 provides the Jaccard Indices of n-gram analysis with 100% sample size when compared to byte frequency algorithm 100% sample size, rate of change 100% sample size and algorithm of Conti et al 100% sample size represented by the comma separated cells respectively.

Table 11 provides the Jaccard Indices of the algorithm of Conti et al with 100% sample size when compared to byte frequency algorithm 100% sample size, rate of change 100% sample size and n-gram analysis 100% sample size represented by the comma separated cells respectively.

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
zip	0, 0, 42.4	0, 0, 49.9	0, 0, 7.7	0, 0, 21	0, 0, 13.2	0, 0, 24.1	0, 0, 24.8	0, 0, 0.2	0, 0, 6.2	0, 0, 14.7
text	13, 55.4, 49.2	0, 0, 7.3	87.1, 99, 83.4	19.8, 18.6, 59.7	0.1, 0, 6	4.3, 2.5, 41.9	2.5, 3.7, 39.2	1.6, 11.1, 18.9	0, 0, 0	16.2, 10.5, 40.3
doc	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
mp4	0, 0, 12.9	0, 0, 17.9	0, 0, 4.1	0, 0, 8.5	0, 0, 0	0, 0, 13.7	0, 0, 8.2	0, 0, 4.2	0, 0, 0.1	0, 0, 9
xls	16.6, 17, 0.8	40.7, 9.3, 8.2	0.9, 14.9, 1.4	31.9, 32.5, 28.6	3.2, 15.9, 46.8	77, 76.7, 55.7	32.5, 33.6, 31.1	8.5, 4, 0.8	2.2, 3.7, 0	11.9, 12.2, 5.2
ppt	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
jpg	0, 0, 13.9	0, 0, 7.9	0, 0, 16.9	0, 0, 14	0, 0, 2.5	0, 0, 6.5	0, 0, 15.6	0, 0, 11.1	0, 0, 11	0, 0, 9.3
ogg	0, 0, 4.7	0, 0, 0.4	0, 0, 1.8	0, 0, 2.9	0, 0, 29.9	0, 0, 2.9	0, 0, 1.6	0, 0, 1.7	0, 0, 35.3	0, 0, 1.5
png	0, 0, 0.1	0, 0, 0.2	0, 0, 0.1	0, 0, 0.1	0, 0, 0.1	0, 0, 0	0, 0, 0.2	0, 0, 0	0, 0, 0	0, 0, 1.2
UnClassified	0, 0, 100	0, 0, 0	0, 0, 100	0, 0, 100	0, 0, 0	0, 0, 100	0, 0, 100	0, 0, 0	0, 0, 0	0, 0, 100

Table 8: Jaccard Indices for BFA (100%)

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0, 24, 6	0, 0, 0.3	0, 0, 0.9	0, 0, 1.9	0, 0, 0	0, 0, 1.2	0, 0, 1.2	0, 0, 1	0, 0, 0	0, 0.1, 0.6
zip	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
text	13, 20.2, 15.5	0, 0, 0	87.1, 87.3, 77.1	19.8, 71.9, 2.3	0.1, 0, 3	4.3, 39.1, 1.2	2.5, 13.9, 3.5	1.6, 12.9, 1.8	0, 0, 0	16.2, 31, 7.9
doc	0, 4.2, 4.3	0, 1.8, 11.5	0, 3.4, 0.2	0, 7.6, 2.9	0, 0, 1.7	0, 1, 6.4	0, 12.2, 11.1	0, 3.9, 10.2	0, 1.4, 14.3	0, 9.9, 3.9
mp4	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
xls	16.6, 30.1, 5.6	40.7, 23.5, 14	0.9, 3.6, 11.7	31.9, 21.1, 23.6	3.2, 75.2, 3.5	77, 97.1, 46.2	32.5, 37.4, 38.3	8.5, 31.3, 2.4	2.2, 21.7, 0.5	11.9, 30.8, 8.5
ppt	0, 70.2, 2.5	0, 94.2, 0.5	0, 84.1, 2.4	0, 74.2, 3.9	0, 44.9, 0	0, 54, 2.4	0, 76.9, 2.3	0, 67.5, 0.7	0, 71.5, 1	0, 86.4, 0.7
jpg	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
ogg	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
png	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
UnClassified	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0

Table 9: Jaccard Indices for ROC (100%)

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0, 24, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0.1, 0
zip	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
text	55.4, 20.2, 70.4	0, 0, 0	99, 87.3, 84	18.6, 71.9, 1.6	0, 0, 0	2.5, 39.1, 0.7	3.7, 13.9, 5.9	11.1, 12.9, 38.7	0, 0, 37.1	10.5, 31, 12.2
doc	0, 4.2, 3.6	0, 1.8, 2.7	0, 3.4, 0.8	0, 7.6, 8.7	0, 0, 0	0, 1, 0.4	0, 12.2, 7	0, 3.9, 1.1	0, 1.4, 0.1	0, 9.9, 1.4
mp4	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
xls	17, 30.1, 7.1	9.3, 23.5, 49.1	14.9, 3.6, 1.8	32.5, 21.1, 17.6	15.9, 75.2, 19.3	76.7, 97.1, 45.5	33.6, 37.4, 59.4	4, 31.3, 5.6	3.7, 21.7, 0.3	12.2, 30.8, 6
ppt	0, 70.2, 0	0, 94.2, 0	0, 84.1, 0	0, 74.2, 0.1	0, 44.9, 0	0, 54, 0	0, 76.9, 0.1	0, 67.5, 0	0, 71.5, 0	0, 86.4, 0
jpg	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
ogg	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
png	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
UnClassified	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0

Table 10: Jaccard Indices for n-Gram analysis (100%)

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0, 6, 0	0, 0.3, 0	0, 0.9, 0	0, 1.9, 0	0, 0, 0	0, 1.2, 0	0, 1.2, 0	0, 1, 0	0, 0, 0	0, 0.6, 0
zip	42.4, 0, 0	49.9, 0, 0	7.7, 0, 0	21, 0, 0	13.2, 0, 0	24.1, 0, 0	24.8, 0, 0	0.2, 0, 0	6.2, 0, 0	14.7, 0, 0
text	49.2, 15.5, 70.4	7.3, 0, 0	83.4, 77.1, 84	59.7, 2.3, 1.6	6, 3, 0	41.9, 1.2, 0.7	39.2, 3.5, 5.9	18.9, 1.8, 38.7	0, 0, 37.1	40.3, 7.9, 12.2
doc	0, 4.3, 3.6	0, 11.5, 2.7	0, 0.2, 0.8	0, 2.9, 8.7	0, 1.7, 0	0, 6.4, 0.4	0, 11.1, 7	0, 10.2, 1.1	0, 14.3, 0.1	0, 3.9, 1.4
mp4	12.9, 0, 0	17.9, 0, 0	4.1, 0, 0	8.5, 0, 0	0, 0, 0	13.7, 0, 0	8.2, 0, 0	4.2, 0, 0	0.1, 0, 0	9, 0, 0
xls	0.8, 5.6, 7.1	8.2, 14, 49.1	1.4, 11.7, 1.8	28.6, 23.6, 17.6	46.8, 3.5, 19.3	55.7, 46.2, 45.5	31.1, 38.3, 59.4	0.8, 2.4, 5.6	0, 0.5, 0.3	5.2, 8.5, 6
ppt	0, 2.5, 0	0, 0.5, 0	0, 2.4, 0	0, 3.9, 0.1	0, 0, 0	0, 2.4, 0	0, 2.3, 0.1	0, 0.7, 0	0, 1, 0	0, 0.7, 0
jpg	13.9, 0, 0	7.9, 0, 0	16.9, 0, 0	14, 0, 0	2.5, 0, 0	6.5, 0, 0	15.6, 0, 0	11.1, 0, 0	11, 0, 0	9.3, 0, 0
ogg	4.7, 0, 0	0.4, 0, 0	1.8, 0, 0	2.9, 0, 0	29.9, 0, 0	2.9, 0, 0	1.6, 0, 0	1.7, 0, 0	35.3, 0, 0	1.5, 0, 0
png	0.1, 0, 0	0.2, 0, 0	0.1, 0, 0	0.1, 0, 0	0.1, 0, 0	0, 0, 0	0.2, 0, 0	0, 0, 0	0, 0, 0	1.2, 0, 0
UnClassified	100, 0, 0	0, 0, 0	100, 0, 0	100, 0, 0	0, 0, 0	100, 0, 0	100, 0, 0	0, 0, 0	0, 0, 0	100, 0, 0

Table 11: Jaccard Indices for algorithm of Conti et al

4.3. Performance

In this section we present the results of the average performance of the four algorithms in terms of execution time. The average time was measured by the total time taken by each algorithm to classify all the fragments divided by the total number of fragments.

Algorithm	Avg. time per fragment (in seconds)
Byte Frequency Algorithm	0.004344
Rate of Change	0.049941
n-Gram Analysis	0.00041
Algorithm of Conti et al	0.085005

From the results we can clearly see that n-Gram Analysis performs the best in terms of performance. It out performs its nearest competitor Byte Frequency Algorithm by a magnitude of 10, Rate of Change by a magnitude of 100 and the algorithm of Conti et al by a magnitude of 200.

5. Analysis of the results

In this section we provide an analysis into the results that we have observed from our experiment. In section 5.1 we analyze the effects of the difference size of the sample data in the detection rates of different file types. Next, in section 5.2, we put the results of classification into perspective by analyzing the true positive detection rates and false positive detection rates. We then move on to compare our results with the results of the original experiments in section 5.3. In section 5.4 we analyze to see if a combination of algorithms could be found that leads to better classification without significant decrease in performance.

We expect that with the use of more sample data, the quality of the fingerprint will increase so will the classification results. We also expect there to be some range in the question of sample data size at which, we could say that adding more sample data to the fingerprint wouldn't affect the fingerprint anymore.

5.1. Effects of different sample data sizes on the detection rates

Looking at the true positive rate of classification results, there isn't any definite pattern that we can identify when increasing the sample sizes. We expected that adding more sample data to the fingerprint would allow for a better classification since more samples of a data type should allow for the fingerprint to better represent the characteristic features of the data type. Even in the worst case scenario, since the larger sample data set included the smaller sample data sets, we would expect the true positive rate to remain unaffected. Only a small number of results actually show such behavior, classification of zip type by byte frequency algorithm, text type by rate of change algorithm, doc type by n-gram analysis and ogg type by algorithm of Conti et al.

In stark contrast to our expectations, the results shows cases where classification results actually deteriorate with addition of more sample data. This can be seem to some extent in the classification results of png type by byte frequency algorithm, pdf type by rate of change algorithm, ppt type by n-gram analysis and jpg type by algorithm of Conti et al.

There are also cases where adding more sample data randomly affects the classification results. This behavior can be observed particularly in xls data type classification by byte frequency algorithm where the first addition to the sample size causes a big decrease in detection rate, adding more sample data causes an increase in the detection rate while the last addition causes the detection rate to increase by more than double the previous value.

We looked into some of the cases where the classification results vary a lot due to the addition of more data in order to examine what was happening. We describe our finding in the following paragraphs.

For the byte frequency algorithm, we noticed a huge change in the classification rate for xls file types when we switched from 10% sample data set to 20% sample data set. The true positive detection rate for xls drops by a lot while the false positive detection rate for text increases considerably. Comparing text type with xls type can be a challenge since the text data are stored as ASCII characters by both types. A possible explanation for this behavior could be that the xls fragments from the test set already contained a lot of text data, but were not classified as text type since the text fingerprint lacked the appropriate values to classify them. This means that when we increased the sample size, the fingerprints were skewed in favor of text type. Further increasing the sample data size however once again increased the detection rate of xls type due to the xls fingerprint being able to identify not only the text data within the xls type but also the other fragments that the text fingerprint isn't able to classify.

In the case of rate of change, the most significant changes in the classification rates upon increasing sample data sizes occur due to confusion between text type and xls and doc types while classifying the text test set. Our argument for before holds in this case as well, since the presence of text data within xls and types, the detection rate for text type improves with added amount of sample data since this time, the absence of non-ASCII data would lead to more fragments being classified as text type.

In the case of n-gram analysis, the algorithm confuses between doc, ppt and xls file types. Since they are all Microsoft Office document types, we would expect their fingerprints to be similar if not exact which would mean it is natural that they are confused with each other. The increase in the false positive detection rate for doc type and the decrease in the true positive detection rate for ppt type could probably be their fingerprint being too similar to each other that the algorithm cannot effectively classify them.

The algorithm of Conti et al also show confusion between xls and text types and also between ogg and mp4, two compressed types. The confusion between compressed types when increasing the sample data could also be answered by their fingerprints being similar enough for the algorithm to be ineffective in classifying them.

5.2. Analyzing the results of the experiment

In section 4.1, we presented the results of our experiments as a confusion matrix showing the true positive rate for the different file types (also referred to as recall as defined below). The purpose of the table was to be able to show at a glance, the detection rates for the different file types. In this section, we would like to measure the overall accuracy of each algorithm as a single number along with the use of the recall and precision value to calculate the F-score for the algorithms. The overall accuracy of an algorithm is calculated as the

$$Overall\ Accuracy = \frac{\sum Number\ of\ true\ positives\ for\ all\ types}{Total\ number\ of\ fragments}$$

The other number that we are interested in is the precision for all the algorithms. The formulas for precision is as follows:

$$Precision = \frac{Number\ of\ true\ positives}{Number\ of\ true\ positives + Number\ of\ false\ positives}$$

$$Recall = \frac{Number\ of\ true\ positives}{Number\ of\ true\ positives + Number\ of\ false\ negatives}$$

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

Algorithm	Precision										Overall Accuracy
Byte Frequency Algorithm	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png	
10% sample data	1.00	0.27	0.42	0.05	0.23	0.51	0.00	0.37	0.28	0.25	0.33
20% sample data		0.26	0.34	0.05	0.26	0.33		0.41	0.29	0.24	0.30
50% sample data		0.26	0.34		0.26	0.47		0.36	0.30	0.25	0.31
100% sample data		0.25	0.45	0.02	0.26	0.57		0.35	0.32	0.23	0.35

Table 12: Precision and accuracy for BFA algorithms

Algorithm	Precision										Overall Accuracy
Rate of Change	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png	
10% sample data	0.77	0.74	0.83	0.58	0.00	0.69	0.13	0.00			0.32
20% sample data	0.87		0.87	0.68	0.00	0.62	0.13	0.00		0.00	0.32
50% sample data	0.92		0.86	0.64	0.00	0.62	0.13	0.00			0.32
100% sample data	0.92		0.87	0.65	0.00	0.68	0.13	0.00			0.33

Table 13: Pericion and accuracy for ROC algorithms

Algorithm	Precision										Overall Accuracy
n-Gram Analysis	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png	
10% sample data	1.00		0.85	0.15		0.59	0.13				0.30
20% sample data	1.00		0.84	0.12		0.60	0.13				0.30
50% sample data	1.00		0.84	0.08		0.58	0.13				0.29
100% sample data	1.00		0.84	0.08		0.60	0.13				0.28

Table 14: Precision and accuracy for n-Gram analysis

Algorithm	Precision										Overall Accuracy
Algorithm of Conti et al	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png	
10% sample data	0.39	0.48	0.36	0.26	0.24	0.44	0.20	0.18	0.36	0.17	0.29
20% sample data	0.39	0.48	0.37	0.27	0.05	0.47	0.19	0.16	0.18	0.14	0.24
50% sample data	0.14	0.48	0.44	0.32	0.00	0.66	0.18	0.16	0.18	0.13	0.29
100% sample data	0.39	0.48	0.44	0.31	0.00	0.66	0.18	0.18	0.17	0.13	0.28

Table 15: Precision and accuracy for algorithm of Conti et al

* The blank cells in the Tables above represent cases where there are no fragments classified in that type by that particular algorithm.

Algorithm	**F-Score**									
	pdf	**zip**	**ext**	**doc**	**mp4**	**xls**	**ppt**	**jpg**	**ogg**	**png**
Byte Frequency Algorithm										
10% sample data	0.00	0.42	0.59	0.01	0.25	0.54		0.16	0.17	0.17
20% sample data		0.41	0.50	0.00	0.25	0.25		0.35	0.17	0.10
50% sample data		0.40	0.50		0.25	0.38		0.34	0.16	0.08
100% sample data		0.39	0.62	0.00	0.26	0.65		0.29	0.16	0.08
Rate of Change										
10% sample data	0.37	0.00	0.73	0.50		0.80	0.22			
20% sample data	0.36		0.79	0.50		0.75	0.22			
50% sample data	0.35		0.80	0.49		0.76	0.22			
100% sample data	0.35		0.85	0.50		0.80	0.22			
n-Gram Analysis										
10% sample data	0.17		0.89	0.12		0.74	0.22			
20% sample data	0.18		0.89	0.11		0.75	0.22			
50% sample data	0.13		0.89	0.10		0.73	0.22			
100% sample data	0.10		0.89	0.11		0.74	0.22			
Algorithm of Conti et al										
10% sample data	0.10	0.46	0.44	0.16	0.37	0.38	0.06	0.23	0.16	0.08
20% sample data	0.08	0.46	0.46	0.16	0.06	0.41	0.06	0.22	0.09	0.07
50% sample data	0.00	0.47	0.57	0.13	0.00	0.55	0.07	0.22	0.11	0.06
100% sample data	0.08	0.46	0.57	0.16	0.00	0.54	0.06	0.20	0.12	0.05

Table 16: The F-Score for all the algorithms

Looking at the F-Scores in Table 16, we still don't seem to have a clear winner as to which is the best algorithm. Rate of Change and n-Gram analysis have almost similar scores, but both seem unable to classify compressed file types. The probable reason for ROC's failure to handle compressed data types may be the lack of too much rate of change between consecutive bytes with high entropy data. Though rate of change rates slightly lower in the text type, it our scores n-Gram in doc and pdf types. On the other hand, as we have mentioned in the previous section n-Gram Analysis is almost 100 times faster than the ROC algorithm per fragment analyzed. Though looking at the timing it may not seem a lot but considering that work in digital forensic requires GB and TB of data to be analyzed, the difference in timing would be over 3.5 hours for only 1GB of data.

n-Gram analysis scores extremely well for text types and xls types and with its advantage in speed, it could be used as a classification algorithm to the two types even though rate of change scores slightly better for xls types. For doc and pdf types, even though rate of change scores are pretty high, its longer processing time could be a hamper in the file carving process, which is already a very time consuming process, thus it might be suitable to compromise some accuracy for much faster speed, but that of course depends on the needs of the analyst.

An interesting thing we note from the F-scores table is that the two faster algorithms, n-Gram analysis and byte frequency algorithm seem to complement each other in finding the chosen file types. In section 5.4, we will take a look at combining these two algorithms to see if we can achieve greater accuracy without compromising too much on performance.

5.3. Comparison of results with original experiments

5.3.1. Byte Frequency Algorithm

A number of differences are present with the way we have tested the Byte Frequency algorithm with how it was originally tested. The most important difference as can be seen from Table 3 is that the original experiment was performed on full files i.e. the fingerprint creation and classification were done with full files. In contrast in our experiments we have used 4KB fragments for fingerprint creation as well as the classification process. Another important difference between the experiments is the number of test data, the original experiment only tested 4 files for each type while we are testing on a much larger set.

Out of the 10 file types that we have chosen for our experiment, 6 of them match with the original experiment. In Table 12, we list the results of the comparison of our experiment with the original experiment.

File Type	Original experiment TP rate	Our experiment TP rate				Overall accuracy of original experiment	Overall accuracy of our experiment			
		10% sample data	20% sample data	50% sample data	100% sample data		10% sample data	20% sample data	50% sample data	100% sample data
						27.50	32.97	30.13	31.13	35.24
doc	0.00	0.50	0.20	0.00	0.00					
jpg	25.00	9.70	30.40	31.80	25.10					
ppt	0.00	0.00	0.00	0.00	0.00					
txt	75.00	96.20	96.30	96.30	96.20					
xls	0.00	56.80	19.90	32.10	76.20					
zip	50.00	86.00	88.00	88.20	89.50					

Table 17: Comparison of BFA results with original experiment

We see that overall our setup of using 4KB fragments for test set and sample set seems to perform on the same level as the original experiment at the worst case which showing significant improvements in the best case. Even though we have presented a comparison between the overall accuracy of the algorithms, we do not feel this is a fair comparison as they difference in test set size is too large (4 files vs 13,365 files). We attribute the increase in the True Positive rate to having a larger sample data set than the original researches. We could say that the method of using 4KB fragments over full files, may have been a cause in the better TP rate but we cannot confirm this and the comparison results of n-Gram analysis later in this section, seems to suggest against it.

5.3.2. Rate of Change

It is hard to make any concrete comparison of our Rate of Change implementation with the original experiment for the following two reasons:

1) They do not provide any concrete numbers for us to compare our classifications against. They provide graphs plotted with detection rates against false positive rates, but we need quantitative data to make concrete comparisons.

2) The focus of the research was identifying JPEG file fragments. They use an extension to the ROC method and present the TP and FP% only for their experiment with ROC and the JPEG extension.

5.3.3. n-Gram Analysis

The difference between the original n-Gram analysis experiment and our work is highlighted by the following two points:

- They use full files for training and for testing they test on different sizes of the files in the test including the full file. We will only compare with the results of the full file because the sizes that the authors have employed all start from the beginning of the file i.e. there are more chances of encountering metadata than actual data.
- All the file types that were actually causing us problems during our experiment were combined into one type as DOC by the authors. We do not afford ourselves this luxury as our aim is to be able to classify fragments for file carving purposes and so this purpose, it does matter whether a fragment was classified as either doc or xls or ppt.

Leaving aside the 3 Mircosoft Office document formats, we still have 2 more file types in common with the original experiment, jpg and pdf. The authors report detection rate of 84% for jpg and 68.3% for pdf files. This is completely different what we have found in our results 0% and 10% respectively at best.

We believe there could be two reasons why there is such a vast difference in the result. One of the reasons could be our use of 4KB fragments instead of full sized files, but the improvement in detection rate with byte frequency algorithm leads us to believe the problem lies in the second reason, i.e. combining the Microsoft Office document files into the doc type. Looking through our data, we see that most of the fragments for both jpg and pdf types have been confused with ppt and doc types. PPT type has a larger influence than doc type. In the absence of the ppt type or combining the documents types into one, we could achieve the percentages mentioned by the author but like mentioned before, for our purposes each file type, as long as it has been identified as a different type has to be tested as such in order to improve the performance of the file carver that we hope these algorithms will help.

5.3.4. Algorithm of Conti et al

Once again we must first explain the difference between the original experiment and our approach. The first difference is that the original experiment used 1KB fragments against the 4KB that we are using. Another difference between our approaches and probably the most significant is that in the original experiment classification was done by polling neighbor using the k-nearest neighbor algorithm. In contrast, we have opted to leave out the polling as when analyzing storage media we are more often than not likely to face situations where the storage of data is not serial but spread across all the chips of a storage media.

The detection rates from the original experiment are better than our results. Most notably, they had 42.4% detection rate for png type, 44.1 for jpg, 98.7 for text while we have 5%, 35.6% and 82.3% respectively at most. Once again, though the use of different fragment sizes pop up as a plausible cause, we still think that they differences in the results are basically due to the classification method we employed instead of using the k-nearest neighbor algorithm. However, we would like to do an experiment with 1KB fragment sizes to confirm whether it is actually better to employ smaller fragment sizes for the classification.

5.4. Possibility of combining algorithms

Looking at the interesting possibility of combining two algorithms that seem to complement each other, in this section we set out to see how the combination would fare with the help of Jaccard Indices that we calculated. For the purposes of simplicity, we have chosen BFA with 100% sample data and n-Gram analysis with 100% sample data.

We try to combine the results of the two algorithms giving us a single classification result which acts almost like we have twice the data than what we tested. Calculating the F-Score for this new classification result actually leads to a lower score than when the two algorithms were individually checked.

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	12909	0	0	0	0	0	0	0	0	1
zip	85638	211119	5409	47654	62803	91	127007	100402	23257	172725
ext	62683	471	455502	120601	9821	57463	14333	4558	42647	6817
doc	79989	13705	1842	38818	123017	182	57944	79619	67866	31595
mp4	26825	7696	808	10295	60747	20	24440	36027	53081	17554
xls	41448	862	2907	171525	1035	416023	46239	10865	10548	8672
ppt	108360	221876	7129	63284	100484	124	157479	156617	168448	202768
jpg	15118	7617	990	8608	19909	7	21442	59754	19441	17154
ogg	38486	4222	456	11435	62655	23	17335	24016	79019	10029
png	3698	3840	328	2329	7995	2	6415	4088	9557	11530
UnClassified	4	0	7	261	0	43	32	0	0	1
F-Score	0.05	0.32	0.73	0.08	0.18	0.70	0.19	0.19	0.22	0.04

Table 18: Combined classification result (number of fragments) with F-Score

This leads us to the conclusion that adding the overhead of an extra algorithm may not necessarily lead to better classification results.

6. Discussion

We realize that our work has been limited to the four algorithms we have used. We would have liked to use more algorithms but the performance issues with running 4 instances for each algorithm quickly creeps up and the total time needed for the analysis increases a lot. The algorithms that we have chosen are some of the less computational heavy algorithms that we have seen and even with these algorithms it takes around 5-10 hours to run one instance of all algorithms over 1GB of data. The case for choosing fast algorithms is to try to make sure that the increase in accuracy for file carving does not bring about a drastic decrease in performance. In case of the need to analyze 1 TB or larger amount of data, we still need algorithms that have better performance. That is what make this problem not only a digital forensic problem but a software engineering one too.

There is also an issue of the limited number of types that we have included in our experiment. There are many file types that we would have liked to include in this experiment but we couldn't for the primary reasons that it isn't easy finding enough data for us to use for the different file types. We could have used search engines to look for different file types but there is a limit to the number of types that search engines can find too. Even if we did find the amount of data we were looking for, the sheer volume of work required to make sure that we wouldn't be violating any copyright laws or licenses in using the files would make the process impossible. Furthermore, we first wanted to see that the results on a limited set of file types were actually good enough to warrant the need for working on a large set. Even with the limited file types we have used, we have seen that classification for file fragments is neither fast nor very accurate and since the results were not as good as we expected, even the limited set of file types has shown that these algorithms are not ideal for file fragment classification.

Another point for consideration would be whether the first 10% of the data set that is used in fingerprint generation of all sample data, was so inconsistent with the data that was tested that it actually biased all the results. As an example, we can look at the Microsoft Office document files that seem to bias all algorithms. The xls, ppt and doc types have the highest classification results combined. If there was a bias created towards these file types due to the files the data that we have used then it wouldn't be a problem with the algorithms and this would void our findings in this experiment. We have intentionally tried to make the data sets as diverse as possible using files from different sources and by employing a large data set we have tried to make sure that such biases, if any, are minimized by the data in other sets.

One of the biggest problems with classifying file types is whether we can say we had a representative set for the algorithms to work on. There we no way, that we know of, that we can prove that we did. With the sheer amount of data out there and the number of different types that can be used in the process of creating a single type, like pdf and doc files can contain text and images and the different little differences in jpg files between vendors of different cameras, it would be impossible to say for certain that we did have a representative file set. The best we can do is make sure that we use as much data as possible, which is what we have done in this experiment. Our use of data from a variety of source and the amount of data we have used allows us to rest assured that we indeed have a very diverse set of data to represent the file types, if even they more be exactly representative sets.

7. Conclusions and Future Works

In this section we present the conclusion we have reached through our experiment in section 7.2 and finally the work that we would like to include in the future in section 7.3.

7.1. Conclusion

We wanted to know if there was a best algorithm for classifying file types. Through the experiment, we have shown that for text and xls types, rate of change and n-gram analysis perform considerably well with scores of 89% and 80% respectively. For the other types however the results from the all the algorithms weren't as good. The best scores we could get were of around 50% for the doc type by the rate of change algorithm. In terms of performance, we see that the n-gram analysis is clearly superior to the other algorithms by a huge margin. This leads us to conclude that if we had to say that there was an algorithm that was the best, it would have to be the n-gram algorithm, not because it provides us with very good results but because it superior performance. File carving is already a very time expensive process, we wouldn't want to add more performance issues to it, so with the results that we have seen, we would use n-gram analysis to get an idea of the type of the fragment while losing the least possible time, since as we can see from the analysis results, there is a possibility of using up hours and days to classify fragments in case we have 1TB of data.

Looking at the final results, we feel compelled to question whether cheap and fast fragment classification methods will actually work. Our results show that we could spend hours analyzing the fragments, but at the end we could still be nowhere near to knowing what file type it was. Our confidence on the classification results would be less than 50%. That would mean that the classification results have almost the same predictive results as a coin toss, random at best.

7.2. Future works

In the near future, we would like to repeat our experiment to include more algorithms and more file types. We would like to see how the other algorithms perform on the same data set that we have used to these 4 algorithms as well as see how all the different algorithms would perform on a new data set which would answer if we had a 10% sample data set that actually affected the results of our experiment. This would also help us to answer the question that has risen from this experiment of whether cheap and fast fragment classification will actually be possible.

Bibliography

[1] L. Arenson and J. van den Bos, "Towards an Engineering Approach to File Carver Construction," in *35th IEEE Annual Computer Software and Applications Conference Workshops*, 2011.

[2] V. Roussev and S. Garfinkel, "File fragment classification - the case for spefcialized approaches," in *Systematic approaches to digital forensic engineering*, 2009.

[3] M. McDaniel and M. Heydar, "Content Based File Type Detection Algorithms," in *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, 2003.

[4] M. Karresand and N. Shahmehri, "File Type Identification of Data Fragments by their Binary Structure," in *Proceedings of the 7th Annual IEEE Information Assurance Workshop*, 2006.

[5] S. Moody and R. Erbacher, "Statistical Analysis for Data Type Identification," in *Proceedings of the 3rd IEEE Internation Workshop on Systematic Approaches to Digital Forensic Engineering*, 2008.

[6] C. Veenman, "Statistical Disk Cluster Classification for File Carving," in *Proceedings of the First Internation Workshop on Computational Forensics*, 2007.

[7] S. Garfinkel, P. Farrell, V. Roussev and G. Dinolt, "Bringing science to digital forensic with standardized forensic corpora," in *DFRWS*, 2009.

[8] "Digital Corpora," [Online]. Available: http://digitalcorpora.org/.

[9] J. van den Bos and T. van der Storm, "Domain-Specific Optimization in Digital Forensics," *Lecture Notes in Computer Science,* vol. 7307, pp. 121-136, 2012.

[10] "Academic Earth," [Online]. Available: http://www.academicearth.org/.

Appendix A

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	1	0	0	0	0	0	0	0	0	0
zip	78649	202746	4679	42688	49507	87	113892	79751	16000	150440
ext	37448	452	228823	113449	10693	102115	13231	2737	24746	5741
doc	5013	16	42	1409	1519	3	978	279	19438	725
mp4	24170	10763	1069	9942	61097	10	29204	59994	43168	27308
xls	27175	764	843	42522	581	134685	25922	10656	15373	4319
ppt	0	0	0	0	1	0	0	0	7	0
jpg	6255	3164	519	4904	453	4	10946	23307	8107	4559
ogg	49085	7170	707	15486	89152	36	25844	38823	95349	15508
png	9779	10629	1000	6744	11230	6	16284	22426	14744	30822
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 19: Classification results for BFA (10%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0	0	0	0	0	0	0	0	0	0
zip	82254	207602	5121	45035	58541	91	120527	83112	20671	165852
ext	40633	497	229113	136957	10431	189432	22448	3017	34172	6134
doc	1263	11	22	533	160	0	332	84	8563	254
mp4	21309	6941	729	8009	52586	12	21803	32010	43575	16888
xls	27385	749	571	19803	637	47372	17314	11016	15970	4306
ppt	0	0	0	0	0	0	0	0	0	0
jpg	15650	9229	1136	10036	13049	8	23916	72453	14912	18476
ogg	44858	6039	602	13988	79469	31	22331	33018	87219	13224
png	4223	4636	388	2783	9360	0	7630	3263	11850	14288
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 20: Classification results for BFA (20%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0	0	0	0	0	0	0	0	0	0
zip	82715	208060	5151	45374	58794	91	121390	85542	21018	166541
ext	47005	723	229093	132683	13375	160723	23696	9531	48401	7525
doc	0	0	0	0	0	0	0	0	0	0
mp4	24046	7015	741	9125	55845	15	22263	32376	49476	16413
xls	22256	617	614	24855	291	76081	16922	5652	9609	3279
ppt	0	0	0	0	0	0	0	0	0	0
jpg	18229	10618	1273	11114	21229	10	27436	75852	21514	23047
ogg	39920	4825	491	11806	66150	26	18348	26229	77884	10771
png	3404	3846	319	2187	8549	0	6246	2791	9030	11846
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 21: Classification results for BFA (50%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	0	0	0	0	0	0	0	0	0	0
zip	85638	211119	5409	47654	62803	91	127007	100402	23257	172725
ext	40311	471	228846	101396	9821	55987	13789	4102	42634	6168
doc	1	1	0	1	0	0	0	0	40	0
mp4	26825	7696	808	10295	60747	20	24440	36027	53081	17554
xls	27498	738	845	55426	303	180816	25873	9584	9903	4262
ppt	0	0	0	0	0	0	0	0	0	0
jpg	15118	7617	990	8608	19909	7	21442	59754	19441	17154
ogg	38486	4222	456	11435	62655	23	17335	24016	79019	10029
png	3698	3840	328	2329	7995	2	6415	4088	9557	11530
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 22: Classification results for BFA (100%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	57389	1775	157	1907	686	14	3531	4519	805	3283
zip	0	53	0	0	2	0	17	0	0	0
ext	9316	0	153846	18972	8	2584	309	35	0	1092
doc	6091	292	30848	104498	407	8065	19966	3184	637	7176
mp4	0	1	0	0	0	0	1	1	0	0
xls	23978	186	44543	22505	325	224847	7491	1092	100	913
ppt	140805	233394	8295	89523	222805	1479	205016	229142	235390	226959
jpg	0	3	0	0	0	0	2	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 23: Classification results for ROC (10%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	54594	592	78	1527	116	12	2195	1910	306	1627
zip	0	0	0	0	0	0	0	0	0	0
ext	6978	0	172684	14364	5	2061	377	28	0	1102
doc	5367	371	3307	92905	139	5564	17017	3334	764	8230
mp4	0	1	0	0	0	0	1	0	0	0
xls	29266	232	53392	42707	598	228417	13003	1628	196	1358
ppt	141374	234486	8228	85868	223375	935	203727	231071	235666	227106
jpg	0	22	0	6	0	0	12	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	28	0	0	1	2	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 24: Classification results for ROC(20%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	52118	191	38	1121	20	12	1353	835	76	952
zip	0	0	0	0	0	0	0	0	0	0
ext	4312	0	178248	19754	1	2273	340	57	0	1100
doc	6973	633	3508	93305	132	4148	19793	4377	1318	11013
mp4	0	0	0	0	0	0	1	0	0	0
xls	31562	407	47740	39251	616	229485	14082	1998	309	1971
ppt	142614	234466	8155	83969	223464	1071	200761	230706	235229	224387
jpg	0	7	0	5	0	0	3	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 25: Classification results for ROC (50%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	51385	122	37	1147	6	15	1258	751	37	821
zip	0	0	0	0	0	0	0	0	0	0
ext	5431	0	199623	21106	18	2423	369	69	0	1100
doc	7005	487	3639	97329	173	5128	20470	4037	1196	10322
mp4	0	0	0	0	0	0	1	0	0	0
xls	30157	433	26221	33715	556	229282	13114	2006	352	1968
ppt	143601	234662	8169	84107	223480	141	201120	231110	235347	225212
jpg	0	0	0	1	0	0	1	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 26: Classification results for ROC (100%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	22275	0	0	5	0	0	2	1	0	1
zip	0	0	0	0	0	0	0	0	0	0
ext	19810	0	224224	17274	0	931	475	418	7	619
doc	41940	4526	723	23851	15491	451	23547	23302	19306	7457
mp4	0	0	0	0	0	0	0	0	0	0
xls	16141	106	4563	114436	730	235446	19628	1129	449	3890
ppt	137413	231072	8179	81839	208012	161	192681	213123	217170	227456
jpg	0	0	0	0	0	0	0	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 27: Classification results for n-Gram (10%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	23878	0	0	5	0	0	1	1	0	1
zip	0	0	0	0	0	0	0	0	0	0
ext	22146	1	226322	17960	0	1321	533	451	12	639
doc	48401	7065	1014	26035	43246	411	29227	29337	26894	9802
mp4	0	0	0	0	0	0	0	0	0	0
xls	13533	102	2282	114740	731	235100	19760	1141	331	4022
ppt	129621	228536	8071	78665	180256	157	186812	207043	209695	224959
jpg	0	0	0	0	0	0	0	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 28: Classification results for n-Gram (20%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	16727	0	0	33	0	0	1	3	0	3
zip	0	0	0	0	0	0	0	0	0	0
ext	21765	0	225806	17613	0	1144	505	429	7	614
doc	66325	11379	1097	31252	103221	99	47921	60383	52993	21639
mp4	0	0	0	0	0	0	0	0	0	0
xls	17058	187	3249	120450	734	235603	21275	1599	989	4749
ppt	115704	224138	7537	68057	120278	143	166631	175559	182943	212418
jpg	0	0	0	0	0	0	0	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 29: Classification results for n-Gram (50%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	12909	0	0	0	0	0	0	0	0	1
zip	0	0	0	0	0	0	0	0	0	0
ext	22372	0	226656	19205	0	1476	544	456	13	649
doc	79988	13704	1842	38817	123017	182	57944	79619	67826	31595
mp4	0	0	0	0	0	0	0	0	0	0
xls	13950	124	2062	116099	732	235207	20366	1281	645	4410
ppt	108360	221876	7129	63284	100484	124	157479	156617	168448	202768
jpg	0	0	0	0	0	0	0	0	0	0
ogg	0	0	0	0	0	0	0	0	0	0
png	0	0	0	0	0	0	0	0	0	0
UnClassified	0	0	0	0	0	0	0	0	0	0

Table 30: Classification results for n-Gram (100%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	13795	1247	968	4978	83	17	6880	3009	1860	2977
zip	35935	103946	403	9831	8261	22	30957	273	2048	24906
ext	14553	35	137385	98265	590	120836	8715	833	25	2461
doc	8391	209	27429	28174	115	36540	4578	1091	318	1405
mp4	53381	64479	3822	27599	184900	33	76475	146509	117321	107270
xls	7578	63	63009	16928	31	79274	10598	177	5	630
ppt	15261	747	787	6908	7	123	7809	2565	1438	3472
jpg	44330	47386	3249	33277	7149	76	70084	76380	69106	76040
ogg	28079	2088	277	4649	20247	8	7571	4497	43297	8079
png	16272	15504	353	6535	2850	17	12634	2639	1514	12182
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 31: Classification results for Conti et al (10%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	10169	938	802	3408	66	8	5036	2254	1192	2141
zip	36693	106577	423	10127	8603	22	31957	289	2182	25843
ext	14884	36	141602	97222	590	116820	8322	846	26	2466
doc	9232	233	25153	27852	43	32858	4949	1233	338	1597
mp4	39710	38370	2229	14759	15985	21	40525	67187	61519	66919
xls	7585	63	61097	18870	104	86989	11270	191	5	661
ppt	16554	891	913	7424	20	115	8289	2739	1988	3793
jpg	47746	71084	4228	37872	11295	81	82953	84662	85488	102163
ogg	30905	13735	852	10451	186738	9	29340	65628	81762	23118
png	24097	3777	383	9159	789	23	13660	12944	2432	10721
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 32: Classification results for Conti et al (20%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	393	15	2	311	0	0	935	843	20	241
zip	36945	107444	427	10243	8724	22	32286	301	2229	26161
ext	21299	44	195811	99297	590	116765	8327	1041	39	2930
doc	10226	409	15552	20127	7	6723	5479	1542	400	1930
mp4	18412	2857	319	3118	4	9	5197	3021	6108	9872
xls	2734	62	16747	25910	142	113218	12147	137	4	385
ppt	24177	1638	1458	9128	84	84	10931	3636	3260	5199
jpg	47060	68974	4148	37604	12119	78	82603	84922	88327	101881
ogg	54507	51754	2892	22761	201988	27	65927	129616	133274	81758
png	21822	2507	326	8645	575	20	12469	12914	3271	9065
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 33: Classification results for Conti et al (50%)

	pdf	zip	ext	doc	mp4	xls	ppt	jpg	ogg	png
pdf	9879	449	479	4314	4	63	5076	1859	649	2273
zip	36446	105832	418	10040	8484	22	31647	283	2152	25539
ext	20261	41	191115	96756	590	112306	7860	993	35	2816
doc	10199	394	20072	25136	6	16906	5440	1471	386	1879
mp4	21481	3204	386	3895	131	13	6466	3796	10984	11348
xls	3169	61	16826	23120	142	107488	12304	133	2	418
ppt	17666	1429	1100	6147	87	27	8140	3269	3105	3753
jpg	32311	31299	2510	26048	5292	58	54954	57817	59748	56650
ogg	68494	90494	4522	34241	208867	46	93806	156695	157461	127099
png	17669	2501	254	7447	630	17	10608	11657	2410	7647
UnClassified	4	0	7	261	0	43	32	0	0	1

Table 34: Classification results for Conti et al (100%)