File Fragment Analysis Using Normalized Compression Distance

Stefan Axelsson, Kamran Ali Bajwa, Mandhapati Venkata Srikanth

Chapter 12

FILE FRAGMENT ANALYSIS USING NORMALIZED COMPRESSION DISTANCE

Stefan Axelsson, Kamran Ali Bajwa and Mandhapati Venkata Srikanth

Abstract The first step when recovering deleted files using file carving is to identify the file type of a block, also called file fragment analysis. Several researchers have demonstrated the applicability of Kolmogorov complexity methods such as the normalized compression distance (NCD) to this problem. NCD methods compare the results of compressing a pair of data blocks with the compressed concatenation of the pair. One parameter that is required is the compression algorithm to be used. Prior research has identified the NCD compressor properties that yield good performance. However, no studies have focused on its applicability to file fragment analysis. This paper describes the results of experiments on a large corpus of files and file types with different block lengths. The experimental results demonstrate that, in the case of file fragment analysis, compressors with the desired properties do not perform statistically better than compressors with less computational complexity.

Keywords: File fragment analysis, file carving, normalized compression distance

1. Introduction

During the course of an examination, a digital forensic practitioner is often faced with a collection of file fragments from slack space on disks, USB flash drives, etc. Sometimes enough metadata (e.g., file links and headers) survives to make reconstruction easy, but often, what exists is a random collection of file fragments that have to be put together to form partially-complete files. Knowing or having an indication of the types of the file fragments (e.g., pictures, executables or text) aids in the reconstruction process. Knowing the file type of the blocks reduces

the number of block combinations that must be attempted to only those corresponding to the specific file type.

Several techniques can be used to identify the file type of a block. This paper focuses on the use of a learning algorithm based on the normalized compression distance (NCD) [9]. NCD is based on the idea that if two compressed data samples are close to the compressed combination of the two samples, then they are more similar than two data samples that, after compression, are far from the compressed combination.

Previous research has applied NCD to file fragment analysis [3, 18]. When developing an NCD file fragment classifier, it is necessary to select the compressor that yields the best performance. This paper investigates which compression performance metrics have the greatest impact on file type classification performance.

2. Related Work

The identification of file types from file fragments is an active field of research [1, 2, 14, 16, 17]. While many different methods have been developed, this section focuses on methods that use machine learning algorithms. It should be noted that the NCD approach performs on par with the best methods for some file types.

The most relevant work is by Veenman [18], who focused on the n-valued problem ($n = 11$) in a 450 MB data set using statistical methods (including a method based on Kolmogorov complexity as is NCD). Veenman reports an overall accuracy of 45%, which is similar to the results of Axelsson, *et al.* [3, 4]. Veenman also developed a set of two-class (cross-type) classifiers that exhibit higher classification accuracy. Calhoun, *et.al.* [6] extended the work of Veenman, but limited their research to four file types that were analyzed in pairs; this complicates the direct comparison of the results obtained by the two efforts.

Cebrian, *et al.* [8] have attempted to identify the most appropriate compressor for NCD. In particular, they evaluated the `gzip`, `bzip2` and `ppmd` compressors and found that the best compressor choice was the one with the highest compression ratio and strongest idempotency. However, the experimental results presented in this paper do not match this finding for the file fragment analysis problem. One hypothesis for the difference is that the input data of Cebrian, *et al.* is several orders of magnitude larger, and their focus is on when the input becomes too large for a compressor to handle (i.e., when the concatenated input is larger than the window size of the compressor, if one exists). Our work investigates the other end of the spectrum where the input is very small; thus, the results obtained are quite different.

3. Normalized Compression Distance

NCD is based on the idea that using a compression algorithm on data vectors both individually and concatenated provides a better measure of the distance between the two vectors. The better the combined vectors compress compared with how the individual vectors compress (normalized to remove differences in length between the sets of vectors), the more similar are the two vectors. The NCD metric is given by:

$$NCD(x,y) = \frac{C(x,y) - min(C(x), C(y))}{max(C(x), C(y))}$$

where $C(x)$ is the compressed length of x, and $C(x,y)$ is the compressed length of x concatenated with y.

In order to apply this metric in supervised learning, it is necessary to select features of the input data used for training. The distances from the features of the training instances to the features of the instances being classified are then computed.

The k-nearest neighbor (kNN) algorithm is commonly used with NCD for classification. In kNN, the k nearest feature vectors are selected and the class of the new sample is assigned according to a majority vote. For example, with $k = 10$, if three of the closest feature vectors are of file type `zip` and seven are `exe`, then the feature vector being classified is assigned to the class `exe` although the closest example might have been a `zip` feature vector.

An advantage of NCD is that anything that can be put through the compression algorithm can be used as a vector [8]. This means that NCD is parameter free and resistant to errors in feature selection. Not having to make any feature selection is a substantial advantage as there are almost an infinite number of ways to select and evaluate features.

4. Research Questions

In order to develop an NCD-based file fragment classifier, a central question is which compression algorithm to use. The determination of the best compressor is best done by evaluating how the candidate compressors perform the task of interest.

Previous research has shown that a compressor that performs well in terms of compression ratio and idempotency performs well for NCD when it comes to other data classification tasks [7, 8, 13]. In the case of compression ratio, a compressor that produces a smaller output is the better choice for NCD classification.

An idempotent operation produces the same result no matter how many times it is applied (barring the first application). In the case of

compression for NCD, the following identities should hold:

$$C(xx) \approx C(x) \quad \text{and} \quad C(\lambda) = 0.$$

The first identity specifies that a desirable compressor can detect that input is repeated and store very little extra information to remember this fact. The second specifies that the compression of an empty input yields an output of length zero. The two identities are unattainable in practice, but they serve as useful benchmarks.

We evaluate three compressors: (i) `gzip` (using the DEFLATE algorithm) [11, 12]; (ii) `bzip2` (using the Burrows-Wheeler transform) [5]; and (iii) `ppmd` (a version of the PPM algorithm) [10]. The three compressors were selected because of their popularity and availability, including open source implementations and algorithm descriptions. Also, they demonstrate the progressive advancement in compression schemes. In fact, both the compression ratio and resource consumption for the three compressors have the following relationship [8]:

$$\texttt{gzip} < \texttt{bzip2} < \texttt{ppmd}.$$

This research evaluates the combined effect of compression ratio and idempotency when used for file fragment analysis on several different block sizes. The reason for considering several block sizes is that different block sizes are encountered in the field, and that it is conceivable that fragment size could affect compressor performance. By exposing the compressor to more data, it could conceivably see more of the file structure, and be better able to classify the fragment.

More formally, the hypotheses posed are:

- **Null Hypothesis (H0):** There is no difference between the idempotencies measured by the compressors.
  Alternative Hypothesis (H1): There is a difference between the idempotencies measured by the compressors.

- **Null Hypothesis (H2):** There is no effect on the measured idempotencies of the compressors by changing block size.
  Alternative Hypothesis (H3): There is an effect on the measured idempotencies of the compressors by changing block size.

Next, we evaluate how the classification performance is affected. We consider the following hypotheses:

- **Null Hypothesis (H0):** There is no difference in the NCD classification performance when a compressor is selected on the basis of compression ratio or idempotency.

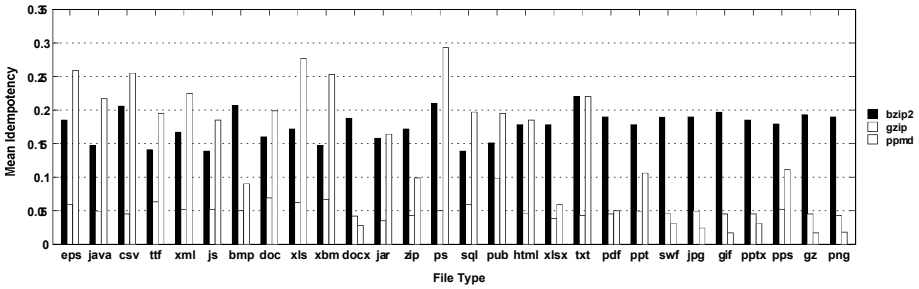

Figure 1. Mean idempotency for all files and all block sizes.

Alternative Hypothesis (H1): There is a difference in the NCD classification performance when a compressor is selected on the basis of compression ratio or idempotency.

- **Null Hypothesis (H2):** There is no difference in the NCD classification performance when the block size is changed.
  Alternative Hypothesis (H3): There is a difference in the NCD classification performance when the block size is changed.

5. Experiments

The testing data contained selected blocks from files in the Garfinkel corpus [15]. The data comprised 50 files of each of the following 28 types: `pdf`, `html`, `jpg`, `txt`, `doc`, `xls`, `ppt`, `xms`, `gif`, `ps`, `csv`, `gz`, `eps`, `png`, `swf`, `pps`, `sql`, `java`, `pptx`, `docx`, `ttf`, `js`, `pub`, `bmp`, `xbm`, `xlsx`, `jar` and `zip`, yielding a total of 1,400 files.

For testing, the files were broken down into block sizes of 512, 1,024, 1,536 and 2,048 bytes. The idempotency of each block was calculated as:

$$NCD(x,x) = \frac{C(x,x) - min(C(x), C(x))}{max(C(x), C(x))} = \frac{C(x,x) - C(x)}{C(x)}.$$

An average for all the blocks of a given file and the average idempotency for all the files of a given file type were calculated for all the block sizes. Figures 1 and 2 show the means and standard deviations of the idempotencies for each of the file types.

Figure 1 shows that `gzip` has better (lower) idempotency values than `bzip2`, which has better values than `ppmd`. Analysis of the standard deviation of the means in Figure 2 reveals that the same pattern appears to hold: `gzip` is more consistent in idempotency while `bzip2` is worse, and `ppmd` is similar to `bzip2`, although it varies wildly for some file types.

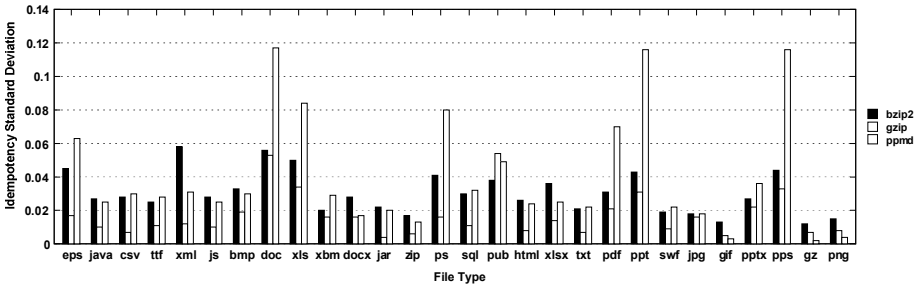

Figure 2. Standard deviations of idempotencies for all files and all block sizes.

We have no scientific explanation for these results. However, one hypothesis is that the fixed overhead plays a part in the compression of what is considered to be a short input by compression algorithm standards.

Table 1. ANOVA test for the null hypothesis.

Source of Variation	SS	df	MS	F	P-Value	F-Crit
Between Groups	0.317	2	0.158	63.46	2.62^{-17}	3.109
Within Groups	0.202	81	0.00249			
Total	0.519	83				

Table 1 lists the results of an ANOVA test on the null hypothesis that there is no difference in idempotencies for the compressors. The null hypothesis is rejected because F is much greater than F-Critical and the P-Value is less than 0.05. Thus, the differences in idempotency for this type of data and for these compressors are statistically significant.

Testing the second null hypothesis that there is no effect on the measured idempotencies when changing the block size involves three cases:

- **Case 1: H0:** Block size original = Block size changed.
  Alt. Hypot. H1: Block size original $\neq$ Block size changed.
- **Case 2: H2:** `bzip2` = `gzip` = `ppmd`.
  Alt. Hypot. H3: `bzip2` $\neq$ `gzip` $\neq$ `ppmd`.
- **Case 3: H4:** All interaction is absent.
  Alt. Hypot. H5: An interaction is present.

Table 2 ("Sample" row) confirms that for Case 1, H0 is valid and that changing the block size has no significant effect on the idempotencies of the three compressors. The "Columns" row shows that for Case 2, H2 is rejected because there is no difference between the idempotencies

Table 2. ANOVA test for the multi-case hypothesis.

Source of Variation	SS	df	MS	F	P-Value	F-Crit
Sample	0.0024	3	0.0008	0.232	0.873	2.63
Columns	0.936	2	0.468	135.388	1.83^{-43}	3.023
Interaction	0.066	6	0.011	3.203	0.0045	2.126
Within Groups	1.12	324	0.00346			
Total	2.125	335				

measured by the three compressors. For Case 3, the "Interaction" row shows that F is greater than F-Critical, but the P-Value is greater than 0.05. Thus, there is tentative support to reject H4, but the result is not significant. In other words, while the experiment measures a difference, the difference might be due to chance or some other factors.

These results are interesting and not obvious. There is clearly a difference in the measured idempotencies for the three compressors with `gzip` having a smaller idempotency value than `bzip2`, whose value is again smaller than the value for `ppmd`. The better the compressor in the general case, the worse its idempotency for file fragment analysis. Furthermore, the idempotency is unaffected by block size, which is less surprising given that 2,048 bytes is a small amount of data compared with the amount of data that these algorithms are designed to handle.

The second set of tests focused on the classification performance of NCD-based file fragment classification given that there are differences in compression performance and idempotency between the compressors. The second set evaluated the `bzip2` and `gzip` compressors on a different selection of file fragments from the same corpus. The `ppmd` compressor was excluded because of its lengthy computational time and because it has a similar compression ratio and idempotency as `bzip2`.

We formulate the following experimental hypotheses:

- **Null Hypothesis (H0):** Although the compressors have different compression ratios and idempotencies, there is no difference in classification performance.
  Alternative Hypothesis (H1): There is a difference in classification performance.

- **Null Hypothesis (H2):** There is no difference in classification performance when the block size is changed.
  Alternative Hypothesis (H3): There is a difference in classification performance.

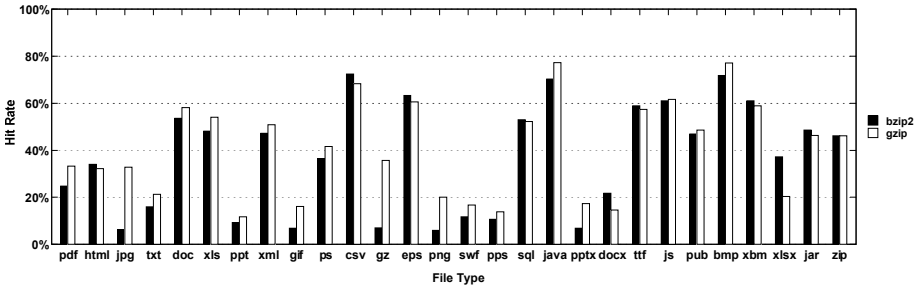

Figure 3. Hit rates for a block size of 512 bytes (average over all k values).

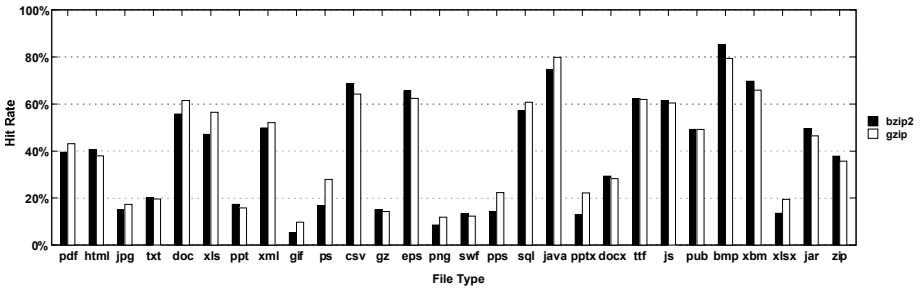

Figure 4. Hit rates for a block size of 1,024 bytes (average over all k values).

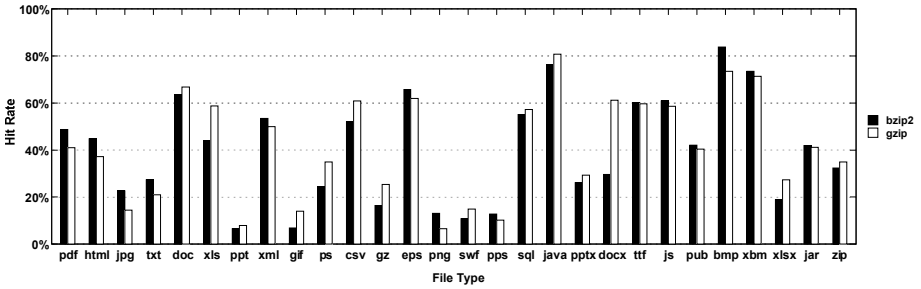

Figure 5. Hit rates for a block size of 1,536 bytes (average over all k values).

This experiment followed the methodology discussed in [3]. The data was divided into ten slices and a form of ten-fold cross validation was performed. kNN classification was then run for each of the ten blocks.

Figures 3, 4, 5 and 6 show the hit rates for various block sizes averaged over all the k values (1 through 10). The figures show similar results for both compressors. While one compressor outperforms the other by a small margin for some file types, the other does the same for other types.

Figures 7 and 8 do not show any clear trends regarding block sizes. The results are similar, although there are differences for certain combinations of block size (e.g., `xlsx` for `bzip2`, but not for `gzip`).

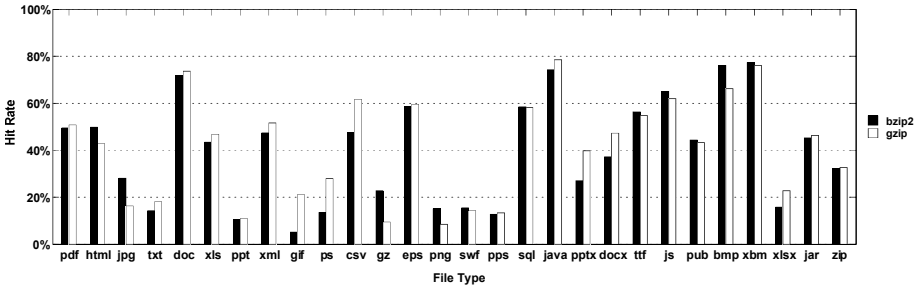

Figure 6. Hit rates for a block size of 2,048 bytes (average over all k values).

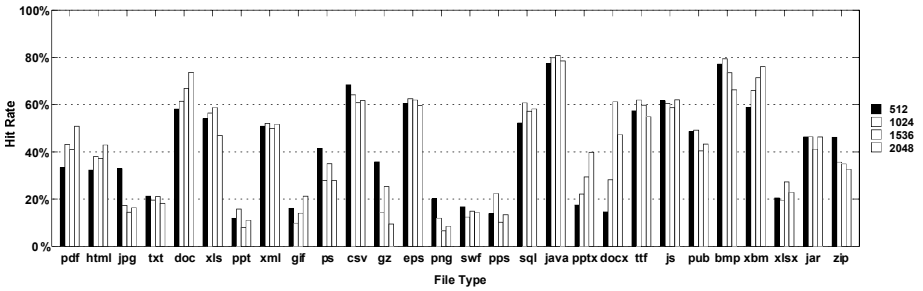

Figure 7. Hit rates for all block sizes (average over all k values, `gzip`).

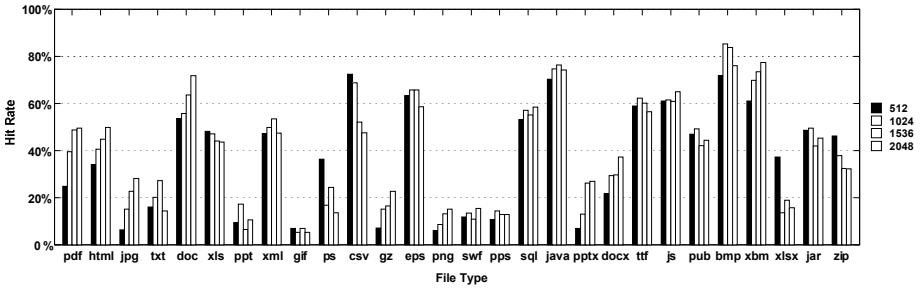

Figure 8. Hit rates for all block sizes (average over all k values, `bzip2`).

An ANOVA test was applied to each hypothesis to test whether or not choosing a compression algorithm based on established measures of effectiveness leads to a better classification outcome. Table 3 shows that there is no statistically significant difference between the classification accuracy of the two classifiers.

The second hypothesis involves multiple cases:

- **Case 1: H0:** Block size original = Block size changed.
  Alt. Hypot. H1: Block size original ≠ Block size changed.

- **Case 2: H2:** `bzip2` = `gzip`.
  Alt. Hypot. H3: `bzip2` ≠ `gzip`.

Table 3. ANOVA test of difference between compressors regarding accuracy.

Source of Variation	SS	df	MS	F	P-Value	F-Crit
Between Groups	2,794.876	19	147.1	0.287	0.998	1.606
Within Groups	277,059.8	540	513.07			
Total	2.125	335				

- **Case 3: H4:** All interaction is absent.
  Alt. Hypot. H5: An interaction is present.

Table 4. ANOVA test of difference between compressors regarding block size.

Source of Variation	SS	df	MS	F	P-Value	F-Crit
Sample	972.3	3	324.1	0.63	0.597	2.609
Columns	6,653.9	19	350.2	0.678	0.844	1.59
Interaction	874.7	57	15.35	0.029	1	1.33
Within Groups	1,115,723	2,160	516.5			
Total	1,124,224	2239				

The corresponding ANOVA test results are shown in Table 4. For Case 1, H0 is retained. The "Sample" row shows that F is less than F-Critical and the P-Value is greater than 0.05. This supports the null hypothesis that there is no significant difference in NCD classification performance of `gzip` and `bzip2` when the block size is changed. For Case 2, H2 is retained because F is less than F-Critical and the P-Value is greater than 0.05. Thus, there is once again no significant difference in the performance when selecting `gzip` or `bzip2`. As in the previous two cases, the null hypothesis H4 is retained for Case 3 – there is no interaction between the two variables.

6. Conclusions

The experimental results demonstrate that there is no significant effect on file fragment analysis accuracy for the NCD algorithm when using different block sizes. Also, there is no significant effect on performance when a compressor is selected on the basis of better compression ratio or idempotency.

Much research remains to be done concerning files and file types (e.g., how to handle a `png` image stored as part of a Word document), especially for the difficult, high entropy files such as already compressed input and encrypted files. Also, since the performance measures of com-

pression ratio and idempotency do not appear to be useful in selecting a compression algorithm, future research should focus on creating new performance measures geared specifically for NCD-based file fragment analysis.

References

[1] I. Ahmed, K. Lhee, H. Shin and M. Hong, On improving the accuracy and performance of content-based file type identification, *Proceedings of the Fourteenth Australasian Conference on Information Security and Privacy*, pp. 44–59, 2009.

[2] L. Aronson and J. van den Bos, Towards an engineering approach to file carver construction, *Proceedings of the Thirty-Fifth Annual IEEE Computer Software and Applications Conference Workshops*, pp. 368–373, 2011.

[3] S. Axelsson, The normalized compression distance as a file fragment classifier, *Digital Investigation*, vol. 7(S), pp. S24–S31, 2010.

[4] S. Axelsson, Using normalized compression distance for classifying file fragments, *Proceedings of the International Conference on Availibility, Reliability and Security*, pp. 641–646, 2010.

[5] M. Burrows and D. Wheeler, A Block-Sorting Lossless Data Compression Algorithm, Technical Report No. SRC-RR-124, Digital Equipment Corporation Systems Research Center, Palo Alto, California, 1994.

[6] W. Calhoun and D. Coles, Predicting the types of file fragments, *Digital Investigation*, vol. 5(S), pp. S14–S20, 2008.

[7] M. Cebrian, M. Alfonseca and A. Ortega, Common pitfalls using normalized compression distance: What to watch out for in a compressor, *Communications in Information and Systems*, vol. 5(4), pp. 367–400, 2005.

[8] M. Cebrian, M. Alfonseca and A. Ortega, The normalized compression distance is resistant to noise, *IEEE Transactions on Information Theory*, vol. 53(5), pp. 1895–1990, 2007.

[9] R. Cilibrasi, Statistical Inference Through Data Compression, Ph.D. Thesis, Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, The Netherlands, 2007.

[10] J. Cleary and I. Witten, Data compression using adaptive coding and partial string matching, *IEEE Transactions on Communications*, vol. 32(4), pp. 396–402, 1984.

[11] P. Deutsch, DEFLATE Compressed Data Format Specification Version 1.3, RFC 1951, 1996.

[12] P. Deutsch, GZIP File Format Specification Version 4.3, RFC 1952, 1996.

[13] R. Feldt, R. Torkar, T. Gorschek and W. Afzal, Searching for cognitively diverse tests: Towards universal test diversity metrics, *Proceedings of the IEEE International Conference on Software Testing, Verification and Validation*, pp. 178–186, 2008.

[14] S. Fitzgerald, G. Mathews, C. Morris and O. Zhulyn, Using NLP techniques for file fragment classification, *Digital Investigation*, vol. 9(S), pp. S44–S49, 2012.

[15] S. Garfinkel, P. Farrell, V. Roussev and D. Dinolt, Bringing science to digital forensics with standardized forensic corpora, *Digital Investigation*, vol. 6(S), pp. S2–S11, 2009.

[16] Q. Li, A. Ong, P. Suganthan and V. Thing, A novel support vector machine approach to high entropy data fragment classification, *Proceedings of the South African Information Security Multi-Conference*, pp. 236–247, 2010.

[17] W. Li, K. Wang, S. Stolfo and B. Herzog, Fileprints: Identifying file types by n-gram analysis, *Proceedings of the Sixth Annual IEEE SMC Information Assurance Workshop*, pp. 64–71, 2005.

[18] C. Veenman, Statistical disk cluster classification for file carving, *Proceedings of the Third International Symposium on Information Assurance and Security*, pp. 393–398, 2007.