Identification of image fragments for file carving

Azzat Al-Sadi, Manaf Bin Yahya, Ahmad Almulhem
Computer Engineering Department
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
E-mail: {g200804300, g201201860, ahmadsm}@kfupm.edu.sa

***Abstract*—Recovering images intact is an important process in digital forensics, as they may represent primary evidences in crime cases such as child pornography. Due to filesyetems' fragmentation mechanisms, images may be split into several fragments on a physical storage. As such, recovering images fragments and reconstructing the original images embody challenges for file carving tools particularly when the filesystem metadata are not available. In this paper, we propose a method for image fragment identification using a machine learning approach. Our method exploits features in unknown images fragments, and apply various machine learning algorithms to reconstruct the original images by identifying to which particular image a fragment belongs. We provide the details of our methods as well as a validation of its effectiveness.**

***Index Terms*—fragments identification, file carving, digital forensics, machine learning**

I. Introduction

A primary task in digital forensics is to extract all available data stored on a digital media as well as recovering usage artifacts such as images, documents, browsing history to name but few [1]. Such files and artifacts in turn may provide crucial evidences leading to solving crimes. Using a filesystem's API and its metadata, it is usually possible to recover most of the files in a digital media. However, in cases where the filesystem is damaged, a different technique is utilized known as file carving [2]. The technique relies on the analysis of raw media data rather than relying on the underlying filesystem [3], [4]. Specifically, files are recovered by identifying their starting and ending signatures.

File carving is relatively straightforward when recovering files stored on contiguous disk blocks. However, in practice, files get fragmented quite often due to various reasons and depending on the used filesystem [5], [6]. Accordingly, carving fragmented files is challenging and requires different heuristics besides matching files signatures. As such, most existing carving tools fail to recover fragmented files.

A general strategy when dealing with file fragments is to perform file carving in two steps. In the first step, the file fragments are grouped into classes of the same file type. For this, content-based analysis is typically used in fragments classification [7], [8]. In the second step, each class is exhaustively searched by trying all the possible combinations in order to reassemble fragments into legitimate files. Some additional information in the file structure such as image markers might be used as we will discuss in section II

In this work, we address the fragments identification problem for image files. Given a pool of image fragments, the objective is to identify all fragments belonging to each file. We propose a machine leaning based method in which features are based on images' pixel values. The obtained results demonstrate the effectiveness of the proposed approach.

The rest of the paper is organized as follows. In section II, we provide a brief review of related work. In section III, we define the image fragments identification problem and some related concepts. In section IV, we provide the details of our methodology, used tools and dataset, and experimentations. Finally, we conclude in section V.

II. Related Works

Garnkel proposed Bi-fragment Gap Carving (BGC) recovery technique [6]. This technique is efficiently used when the number of file fragments is very small (maximum three fragments) and those fragments are not separated by a huge gap. BGC applies an exhaustive search on all blocks combinations between the header and the footer of the JPEG file. Starting with one block gap size, BGC increases the gap between the end of the first fragment and the beginning of the second fragment by one block iteratively. BGC decoded the combined blocks at the end of each search-iteration until successfully validate a recovered file. However, exhaustive search increases time complexity of the BGC technique. Moreover, the false positive rate of BGC may be high because of the decoder errors.

To overcome the decoder validations drawbacks, Pal et al. improved BGC technique by adding another level of validation which is the sequential hypothesis test [9] . After each successful decoding operation, Pal et al. technique scans all consecutive blocks of fragment to determine if these blocks can be joined together. This is done by detecting any block which does not belong to the actual fragment.

In contrast to Pal et al. approach [9] which adds another level of validations, Li et al. improves the decoder to detect fragmentation of corrupted JPEG images [10]. Li et al proposed three techniques using the libjpeg library. Each technique determines the unusual changes in one of the main JPEG image components, namely DC coefficients, AC coefficients and edges along the block boundaries. The combination of these techniques is used to build a detector. Although this detector achieved low false positives with reasonable false negatives, it can be combined with Pal et al. [9] technique

to increase the accuracy of detection. These methods use exhaustive search over all consecutive blocks to detect if there is any block that does not belong to the actual fragment. However, it does not assert whether a fragment belongs to a specific file.

Another technique of reassembling the fragments of non-differential Huffman entropy coded JPEG image was proposed by Karresand et al. [11]. This technique utilizes one of the JPEG file format details which is the restart markers (RST). Unlike the start marker and end marker which appear at the beginning and end of a file, the restart marker (RST) periodically appears in the JPEG file. The data between the RST markers are called Minimum Coding Unit (MCU). The technique uses RST pattern to stop the scan at specific intervals. Moreover, it builds the DC value chains of the JPEG image using the luminance DC values of all restart intervals. Using two sliding windows, the similarity of the DC values chains are measured. The similarity changes of the DC components identifies if the fragments are correctly connected. Unlike our approach, this technique does not use machine learning. Instead, DC coefficients are exclusively used to reassemble the fragments.

Pal and Memon expressed the problem of reassembling file images without the need to validate or decode the reassembled image file [12]. A k-vertex disjoint graph is used to formulate the problem, where k is the number of known base-fragments. Vertices in the graph represent clusters in the unallocated space, and weighted edges represent likelihood that one block (cluster) follow another by utilizing a matching metric produced by examining the pixel boundary after joining blocks. Then reassembly is performed by finding the best possible order of fragments using greedy heuristics.

Mohamad and Deris examined specific fragmented JPEG images scenarios, where the fragmentation point is within the Define Huffman Table (DHT) segment [13]. According to the authors, a fragmentation point at DHT is very important because images could be corrupted when they pass through a decoder. Three possible fragmentation points within DHT area are listed and corresponding three algorithms were proposed to detect these points. Detecting the fragmentation points correctly leads to more accurate image carving. One of the possible scenarios is that a fragmentation point could be placed after the DHT marker. This point will be detected by a validator if the DHT length value is less than 20, because the expected DHT length must be greater than 19. Our approach is different from the work in [9] and [13] where fragmentation point detection precede reassembling the files.

Cohen utilized the detailed structure of JPEG files with mapping function to identify fragmentation point [14]. This technique enhanced the fragmentation point detection by applying exhaustive search using edge detection algorithm. The edge detection algorithm checks the average pixels values of the pixels that surrounding the boundary of the already decoded blocks and the existing blocks.

Instead of using exhaustive search for the remaining file fragments which may reduce the performance of the identification method, Sencar and Memon proposed a bit pattern matching technique to detect the remaining fragments of a given file [15]. Since their technique was applied on JPEG images, they generated the bit pattern from the JPEG Huffman code word. However, the other JPEG components such as markers and parameters are not suitable for the pattern generation. In addition, Sencar and Memon present different methods to solve two challenges in recovering fragments procedures. First, they use the JPEG restart marker to recover the disrupted fragments where the header is not available for these fragments. Then, they generate the proper header from the partially recovered fragments of the same file using their bit pattern technique. Second, they recovered the stand alone file fragments where the header of the whole JPEG file is missing. They utilized the information from other related JPEG files which may be downloaded from the same website or edited by the same software to generate pseudo header.

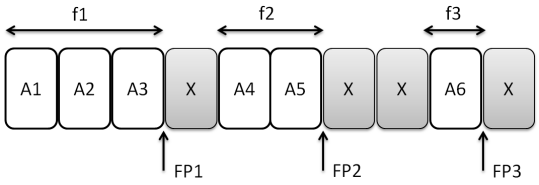

Fig. 1. An example of a fragmented file. A file (A) is fragmented into three fragments (f1, f2, f3) and there are 3 fragmentation points.

III. Image Fragments Identification Problem

To define the problem of image fragment identification, we describe here some related concepts. In Figure 1, a file (A) consists of six blocks (A1 to A6). This file is broken into three fragments (f1, f2, f3) stored in non-contiguous blocks on the hard disk. The first fragment (f1) consists of three blocks and contains the header information of file (A). To recover the file correctly, the first fragment which contains the header information need to be identified. The second fragment (f2) consists of two blocks. The third fragment is (f3) occupies one block and may contain the footer information of file (A). Random data or data of another file may separate these fragments.

A fragmentation point (FP) refers to the last point of the fragment which indicates the end of fragment and the start of a new fragment. As shown in Figure 1, the last points of (A3, A5 and A6) are considered fragmentation points. The detection of fragmentation points is a major issue in file carving. Another major issue concerns fragment identification which refers to associating a given fragments to a specific file and determining its correct order. For instance in Figure 1, the fragment identification problem is to assert that the fragments f1, f2 and f3 belong to file (A) and to assert the correct order.

In this work, we assume that all fragmentation points are detected correctly, and we are left with the fragmentation identification problem. Figure 2 shows a simplified example of the addressed problem. In this example, there are two images (Image1 and Image2) which are fragmented into four fragments. After applying a fragmentation points detection,

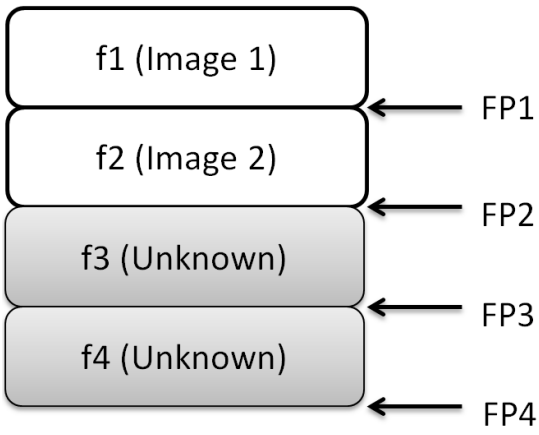

Fig. 2. Illustrating the image fragmentation identification problem

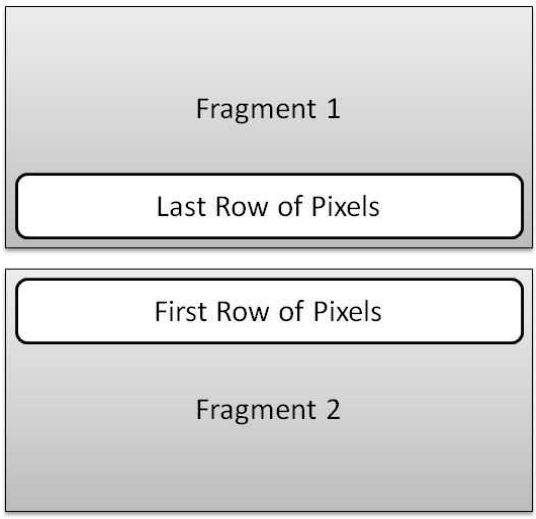

Fig. 3. Comparing pixels' values in a fragment last row with another fragment first row

assume that the four fragments are correctly identified. Using header information, it is possible to determine that the first fragment of Image1 is (f1), and the first fragment of Image2 is (f2). The problem is to associate the remaining two fragments (f3 and f4); i.e. to determine to which image each fragment belongs. Our proposed approach addresses this problem utilizing features of image fragments and machine learning techniques.

IV. Methodology and Experiments

A. Methodology

One property of images is that adjacent pixels have similar values. As such, we propose to employ the values of the adjacent pixels as features. Specifically, the pixels' values of an image fragment's last row is compared with those of the first row of another fragments as shown in Figure 3. We expect that matching fragments to show maximum similarities.

In the proposed approach, last row of all fragments are used for training the classifiers. The first row of all fragments comprises the testing set. We excluded the first fragments as they contain header information. Machine learning techniques are used to evaluate the similarities between pixel values.

Fig. 4. Sample images from the UCID dataset

B. Tools and Datasets

To evaluate our approach, we conducted several experiments using MATLAB R2010a and Weka 3.6.9. MATLAB is used to process images and generate image fragments, while Weka is used to experiment with several machine algorithms.

We used the UCID uncompressed color image dataset (version 2) [16]. UCID comprises a standard images dataset for content based image retrieval (CBIR) research, and is available at [17]. It consists of 1338 vacation photos in uncompressed form taken by digital camera. The dataset have variety of indoors and outdoors images. So, it suites testing our method as it represents typical real life images which an investigator may find in typical users' hard disks. Examples of the images included in this dataset are shown in Figure 4.

Initially, the UCID images are converted into colored and gray-scale jpeg images using MATLAB. Then 50 images of the same dimension (384x512) are randomly selected from the UCID dataset. The same dimension is used to avoid the effect of an image dimension in our experiments. Next, each image is split into three fragments following studies which show that files are rarely split into more than three fragments in practice [6]. The sizes of these fragments are selected randomly.

Then, the training set is populated using the last row's pixels values of the randomly selected 50 images except the last fragment. This results into 100 instances for the training data. Similarly, the testing set is populated using the first row's pixels values of the randomly selected 50 images except the first fragment as it contains header information. This also results into 100 instances for the testing data.

C. Experiments

Using the prepared training and testing sets, we conducted our experiments using the Weka tool. Weka is a widely used tool that implements several popular machine learning algorithms. To evaluate our proposed method, we selected four classifiers, namely NaiveBayesMultinomialUpdateable, MultiClass, RandomForest and BayesNet classifiers. These classifiers are selected due to their popular usage with images.

We proceeded as usual by training the classifiers using the training set, then the testing set is used to find out the detection rate. The detection rate represents the ability of the machine learning algorithm to detect the similarities of pixel values, to identify the unknown fragments and reorder them correctly.

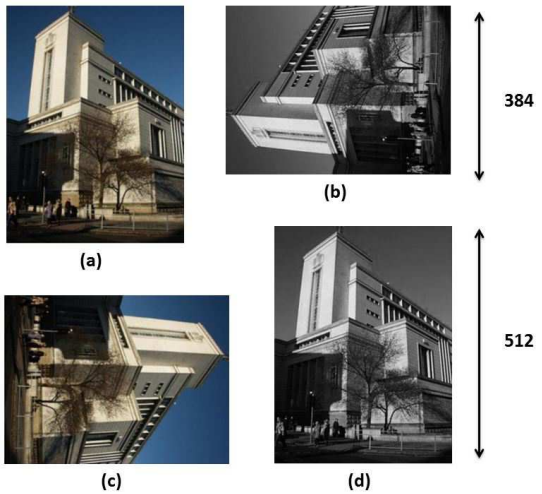

Fig. 5. A sample of tested images: (a) colored image (b) colored rotated image (c) gray-scale image (d) gray-scale rotated image

The experiments are repeated using gray-scale, RGB images, or rotated images to simulate various scenarios. A sample of tested images are shown in Figure 5.

In another set of experimentations, the training and testing set are combined. Then, we re-run the selected classifiers and applied a 10 folds-cross validation. In this experiment the features file is partitioned into 10 equal subsets, one of the subsets is used as a testing set, whereas the other subsets are used as the training sets.

All experiments are repeated five times over the same 50 images, but with different fragment sizes. As such, the fragmentation points of all images differ in each experiment. The detailed results of our experimentations are listed in Tables I, II, III and IV, where C.V. rows refers to the cross validation tests. The columns (Test1 - Test5) correspond to the five runs, and list the percentages of correctly classified instances in each run. The last column lists the average percentages.

The results show that using pixels' values as features for image fragments identification is indeed effective. As shown in Table II, the BayesNet algorithm gives the best results with an average up to 99.2% in identifying and reconstructing an image file from its fragments. In some tests, perfect results (100%) are achieved as shown in Table I. Apparently, the BayesNet classifier seems to give the best results in our experiments. Also, the results indicates that colored image fragments are identified better than grayscale ones.

TABLE I
EXPERIMENT RESULTS OF NAIVEBAYESMULTINOMIALUPDATEABLE ALGORITHM

Images	Test1	Test2	Test3	Test4	Test5	Avg.
Colored	99%	99%	97%	100%	97%	98.4%
Colored Rotated	91%	96%	96%	93%	94%	94%
Grayscale	98%	96%	98%	100%	98%	98%
Grayscale Rotated	88%	89%	94%	95%	92%	91.6%
C.V. Colored	96%	99%	96.5%	98.5%	93%	96.6%
C.V. Grayscale	97.5%	95%	98.5%	96%	98%	97%

TABLE II
EXPERIMENT RESULTS OF BAYES NET ALGORITHM

Images	Test1	Test2	Test3	Test4	Test5	Avg.
Colored	99%	99%	100%	99%	99%	99.2%
Colored Rotated	99%	99%	100%	98%	100%	99.2%
Grayscale	98%	98%	96%	96%	97%	97%
Grayscale Rotated	96%	97%	99%	98%	99%	97.8%
C.V. Colored	87.5%	96.5%	89.5%	94%	87.5%	91%
C.V. Grayscale	94%	92.5%	94.5%	97.5%	93%	94.3%

TABLE III
EXPERIMENT RESULTS OF RANDOM FOREST ALGORITHM

Images	Test1	Test2	Test3	Test4	Test5	Avg.
Colored	85%	82%	83%	89%	87%	85.2%
Colored Rotated	86%	80%	86%	77%	86%	83%
Grayscale	76%	80%	79%	80%	70%	77%
Grayscale Rotated	79%	78%	82%	85%	84%	81.6%
C.V. Colored	76.5%	79.5%	76.5%	80%	77%	77.9%
C.V. Grayscale	75%	77%	77.5%	78.5%	79.5%	77.5%

TABLE IV
EXPERIMENT RESULTS OF MULTICLASS CLASSIFIER ALGORITHM

Images	Test1	Test2	Test3	Test4	Test5	Avg.
Colored	91%	90%	90%	89%	91%	90.2%
Colored Rotated	82%	88%	90%	89%	92%	88.2%
Grayscale	89%	82%	94%	91%	87%	88.6%
Grayscale Rotated	86%	80%	83%	77%	86%	82.4%
C.V. Colored	87.5%	88%	84%	89.5%	84.5%	86.5%
C.V. Grayscale	75%	77.5%	89%	86.5%	86%	82.8%

V. CONCLUSION

Extracting every possible data on a digital media is an important task in digital forensics investigation. File carving is a powerful technique that is often used when the underlying filesystem of a digital media is damaged. This technique is relatively straightforward if recovered files are not fragmented. However, in practice, fragmentation does occur due to various reasons including media performance and daily use. Fragmentation complicates the carving process as recovered fragments need to be associated with specific files. In this work, we addressed the problem of image fragments identification. Specifically, the problem is to associate each image fragment in a given pool of image fragments to a specific image file. Our approach utilizes image pixels values as features and adopts a machine learning approach. The obtained results show the effectiveness of the proposed approach.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia and the Hadhramout Est. for Human Development in Yemen during this project.

REFERENCES

[1] M. Simon and J. Slay, "Enhancement of forensic computing investigations through memory forensic techniques," in *International Conference on Availability, Reliability and Security, 2009. ARES '09*, 2009, pp. 995–1000.

[2] A. Pal and N. Memon, "The evolution of file carving," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 59–71, 2009.

[3] Golden G. Richard and Vassil Roussev, "Scalpel: A frugal, high performance file carver," 2005.

[4] "Foremost." [Online]. Available: http://foremost.sourceforge.net/

[5] Kryder and Mark, "Future storage technologies: A look beyond the horizon," 2006.
[6] S. L. Garfinkel, "Carving contiguous and fragmented files with fast object validation," *Digital Investigation*, vol. 4, Supplement, pp. 2–12, Sep. 2007.
[7] C. Veenman, "Statistical disk cluster classification for file carving," in *Third International Symposium on Information Assurance and Security, 2007. IAS 2007*, 2007, pp. 393–398.
[8] V. Roussev and S. Garfinkel, "File fragment classification-the case for specialized approaches," in *Fourth International IEEE Workshop on Systematic Approaches to Digital Forensic Engineering, 2009. SADFE '09*, 2009, pp. 3–14.
[9] A. Pal, H. T. Sencar, and N. Memon, "Detecting file fragmentation point using sequential hypothesis testing," *Digital Investigation*, vol. 5, Supplement, pp. S2–S13, Sep. 2008.
[10] Q. Li, B. Sahin, E.-C. Chang, and V. Thing, "Content based JPEG fragmentation point detection," in *2011 IEEE International Conference on Multimedia and Expo (ICME)*, 2011, pp. 1–6.
[11] M. Karresand and N. Shahmehri, "Reassembly of fragmented JPEG images containing restart markers," in *European Conference on Computer Network Defense, 2008. EC2ND 2008*, 2008, pp. 25–32.
[12] N. Memon and A. Pal, "Automated reassembly of file fragmented images using greedy algorithms," vol. 15, no. 2, pp. 385–393, 2006.
[13] K. M. Mohamad and M. M. Deris, "Fragmentation point detection of JPEG images at DHT using validator," in *Proceedings of the 1st International Conference on Future Generation Information Technology*, ser. FGIT '09. Berlin, Heidelberg: Springer-Verlag, 2009, p. 173180.
[14] M. I. Cohen, "Advanced JPEG carving," in *Proceedings of the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop*, ser. e-Forensics '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 16:116:6.
[15] H. T. Sencar and N. Memon, "Identification and recovery of JPEG files with missing fragments," *Digital Investigation*, vol. 6, Supplement, pp. S88–S98, Sep. 2009.
[16] G. Schaefer and M. Stich, "UCID: an uncompressed color image database," pp. 472–480, Dec. 2003.
[17] "Ucid [uncompressed colour image database] v2." [Online]. Available: http://cdb.paradice-insight.us/?corpus=40