Journal of Digital Forensics, Security and Law

Manuscript 1772

AN ML BASED DIGITAL FORENSICS SOFTWARE FOR TRIAGE ANALYSIS THROUGH FACE RECOGNITION

Gaurav Gogia

Parag H. Rughani

AN ML BASED DIGITAL FORENSICS SOFTWARE FOR TRIAGE ANALYSIS THROUGH FACE RECOGNITION

Gaurav Gogia[1], Parag Rughani[1]
[1]National Forensic Sciences University, Gandhinagar, Gujarat, India;
{gaurav.phdcs21, parag.rughani}@nfsu.ac.in

ABSTRACT

Since the past few years, the complexity and heterogeneity of digital crimes have increased exponentially, making digital evidence & digital forensics paramount for criminal investigation and civil litigation cases. Some of the routine digital forensic analysis tasks are cumbersome and can increase the number of pending cases, especially when there is a shortage of domain experts. While the work is simple, the sheer scale can be taxing. With the current scenarios and future predictions, crimes will only become more complex, and the precedent of collecting and examining digital evidence will only increase. In this research, we propose an ML-based Digital Forensics Software for Triage Analysis called Synthetic Forensic Omnituens (SynFO) that can automate evidence acquisition, extraction of relevant files, perform automated triage analysis and generate a basic report for the analyst. This research shows a promising future for automation with the help of Machine Learning.

Keywords: digital investigation; digital forensics; machine learning; face recognition, automation

1. Introduction

With the availability of high-speed internet and larger data storage, the growth in the accessibility of internet and internet connected devices has increased exponentially in recent years. These interconnected devices have gained popularity due to their ability to enable users to connect, communicate and share information in various ways. Digital Forensics, a process of investigating crimes with the help of science and technology, is preferred by most investigating agencies worldwide. The only way to perform digital forensics investigation is through the use of some software/hardware tool, tasks including evidence acquisition, extraction of files, and analysis all require some digital forensics software [1]. The digital forensic process includes various steps ranging from evidence acquisition, imaging/copying storage media, analysing the copy of storage media, finding relevant artefacts, interpreting artefacts, creating a timeline of the incident, concluding an investigation based on the found artefacts and finally, preparing a report to be presented in front of the intended parties. It has been observed that the whole process is highly time consuming and requires many person-hours even with the availability of tools that can automate certain processes. Due to confidentiality and the sensitive nature of work, there is little room for errors which is also one factor that investigating agencies can only partially rely on automated tools.

Though these tools are designed to ease the majority of investigators' tasks by automating known procedures, they still need to be capable of addressing many routine tasks like analysing hours of CCTV footage. These limitations cost valuable human hours as investigators need to watch or search multimedia files to find a clue related to the suspect or victim. With technological advancement and the discovery of modern techniques like Machine Learning, there is always scope to add value to the existing forensic investigation tools. This work focuses on the application of Machine Learning in improving the digital forensic process. It especially solves the problem of volume and visually identifiable artefact detection in pictures and videos. We propose an ML based Digital Forensics Software that combines and automates the acquisition of digital evidence, extraction of relevant files (including but not limited to pictures), triage analysis of pictures through person of interest identification, and generation of reports with a single command. The upcoming part of the related work section plunges deep into CCTV. However, this paper's or the software's aim is wider than CCTV forensics. These studies are only meant to provide a road toward the importance of multimedia forensics within the digital forensics domain and how SynFO plays its role in contributing to multimedia forensics and its triage analysis capabilities.

One of the applications for these interconnected devices is the network of CCTV (Closed-Circuit Television), which has drastically changed the surveillance system. Further, the addition of cameras in smart/mobile phones has enabled users to capture and store photographs and videos from the phone itself. Due to their evidential value, the pictures taken by the users or footage captured by a public CCTV have become very important in investigating crimes [2]. As observed in recent studies, the majority of crimes today range from conventional crimes like murder, theft, accident, kidnap, and others to digital crimes like fake profile creation, forged digital documents and published abusive content. Others require investigation of either CCTV footage or multimedia files found from the suspect or victim's device(s), like computers and smartphones [3].

The rest of the paper is structured with sections "Related Work" 2, which presents existing use cases of automation in digital forensics and how it impacts data retrieval, analysis, and data enrichment. The "Methodology" 3 section discusses the process and algorithm involved in developing this software, including programming languages and libraries used during the development phase. The methodology subsection "Automated PoI Identification" 3.4 discusses an empirical method that applies deep learning and computer vision algorithms, which are evaluated in the aftermentioned test cases. The next section, "Presentation" 4 demonstrates the use of SynFO CLI software before moving forward with discussion in "Results and Discussions" 5 section. Post results discussion, this paper reviews "Limitations and Future Scope" 6, this study is wrapped up in the end with "Conclusion" 7 section.

2. Related Work

A review of existing studies suggests the use of artificial intelligence and machine learning has been experimented with by researchers worldwide in multiple scenarios, including but not limited to object detection, natural language processing, triage analysis, etc. The idea revolves around achieving automation to skip past physically taxing manual, labour-intensive tasks [4] by implementing machine learning or rule-based intelligence [5]. Some problems that can be addressed through machine learning and artificial intelligence algorithms include artefact classification [6], timeline reconstruction [7], artefact correlation [8], outlier detection [9], testing automated forensic analysis engines [10], and forged document detection [11]. The machine learning based digital forensics software developed in this study attempts to solve the last problem, detecting forged documents through face verification as one of the modules.

Document forgeries are not limited to just doctored images; another form of forgery involves face morphing. Automatic generation of morphs was achieved by utilizing Benford features calculated by quantized DCT coefficients of JPEG-compressed images. A linear Support Vector Machine is trained on Benford features and is used to detect morphed faces in images by [12] automatically. Face morphing is a common cybercrime; there are other methods for detecting morphed faces. One such method relies on continuous image degradation. This degradation approach creates multiple artificial self-references that eventually help classify faces in an image, as stated by [13]. A deep learning-based method also exists for forgery detection that uses pre-trained Convolutional

Neural Networks, which feeds a Support Vector Machine trained on RGB images' hierarchical representations to help detect image forgeries [14].

Images have become one of the most mainstream methods of sharing ideas and communicating thoughts; the proliferation of image-sharing applications has made it easy for anyone to find a dataset of facial images, download them and utilise them for any use case. Therefore, it is essential to identify whether the given image is original or processed. These processes can be identified by a multi-class classification neural network that identifies whether an image has been processed and can classify the most common post-processing methods [15]. While image forgery detection is not one of the goals, existing studies suggest ruling out doctored images before sending them through Person of Interest identifiers can help against false positives.

Identifying a camera's make and model can turn the tables during case examination. Images from different cameras can be sent through a multi-class SVM classifier based on extracted features that can help identify the camera. Authors [16]could achieve more than 95% accuracy with their database. This work can lead to ease in PoI (Person of Interest) identification by connecting images to the camera that they were shot. Moving forward with images and document forgery, a need to identify fake faces, facial morphs, and facial recognition, in general, has become paramount in forensic investigation. With the advancement in image processing, machine learning, and artificial intelligence, it is now easier to create realistic looking fake faces by employing Generative Adversarial Networks (GAN) or skilled graphic designers. Research by [17] proposes a neural network that helps detect fake or generated images by humans or machines without resorting to metadata information. Similarly, a novel approach based on supervised deep learning can help classify original and retouched images. This helps detect retouched images with software, detect makeup in face images, and distinguish between original and modified images [18].

Some of the use cases where computer vision related algorithms can help in digital forensics include the detection of child pornography, child sexual abuse, and related crimes. Another research that focuses on features based on iris geometry helps automatically detect children in images. This approach proposes the Face to Iris Area Ratio (FIAR), which is based on the fact that iris size remains almost constant in childhood. The algorithm provides a robust method for identifying underage children in digital images [19]. A framework for testing different cameras for their video quality, forensic value, and scope for face recognition from the videos produced by CCTV cameras was proposed by [20]. The framework helps identify ideal camera resolution and distance for higher forensic value and face recognition accuracy. Similarly, a comprehensive comparative study on different AI algorithms was done by [21].

Much work has been carried out to improve automation in this niche field of digital forensics. Sometimes through rules and configuration and other times through machine learning and artificial intelligence implementation. However, the proposed solutions are mostly independent modules, and it may be a challenging task to integrate them with existing digital forensic tools. In this article, we offer a single tool with acquisition and analysis methodology that will help speed up a forensic investigation using state-of-the-art machine learning and computer vision algorithms along with some rule-based automation for data acquisition, file-type identification, and person of interest identification.

3. Methodology

Synthetic Forensic Omnituens (SynFO), is an ML based digital forensics software. The software is designed to automate and streamline forensic triage analysis by combining evidence disk image creation, integrity calculation, relevant artifact extraction, and Person of Interest (PoI) identification in a single command, hence saving many work hours which would be otherwise wasted by manually sifting through thousands of potentially unrelated images.

This section is responsible for answering two broad questions:

1. How was SynFO software developed?
2. How does SynFO software work?

How was SynFO software developed?
This work focused on developing an integrated utility based on open-source technologies. It was decided to use GoLang & Python for writing the software due to their unique features in solving different problems. GoLang [22]was selected for the systems programming part due to its simple learning curve, fast execution, and compilation speed that allows for writing efficient and reliable software. GoLang also has the smallest memory footprint, which is another added advantage. Further, GoLang's concurrency model, fast runtime, and simplicity set it apart from other languages [23]. Python [24], on the other hand, was selected due to the plethora of available libraries for data science related tasks. Popular libraries like DLIB and face_recognition, along with the wide acceptability of Python in the community, made it an easy choice [25]. The final product was tested on Linux & Unix based platforms (Arch Linux & MacOS).

The design goal of the tool was to follow this algorithm:

1. Creates an image/copy of the acquired evidence (storage media)
2. Analyses the image created in the previous step to extract specific types of files based on the option set by the user (it can parse, reconstruct and recover files automatically)
3. Performs one of the following tasks based on the option selected:
    a. Automatic PoI Identification
    b. Store extracted files from the disk image
4. Generates results

How does SynFO software work?
To answer this question, this section explains how the proposed tool SynFO works at different stages of the digital forensics process, namely acquisition, extraction, and automated person of interest identification.

3.1. Acquisition

The first step of the tool SynFO is to create a disk image of the storage device under investigation. This disk image is created by copying all the data from the device file of the connected storage media in question. This device file is usually located in /dev/ directory in Linux/Unix based machines. A storage device in question is unmounted before the copy process begins to avoid accidental writes on the device. SynFO reads raw bytes from the source device file and creates a bit-stream copy of the same. The default buffer size is set to 10240 bytes, but that can be changed via command line flags. After the bit-stream copy, a single image file is written on the disk. This entire imaging process is timed, and an elapsed time is presented in a human readable format, so the time for imaging can be documented. As the standard procedure requires, the software calculates MD5 and SHA256 hashes. These hashes help maintain the integrity of the evidence through the investigation process. To begin the process, the investigator can connect the device and run the SynFO through the command line. The following section explores the second step in the SynFO process.

3.2. Extraction

The second step of SynFO is to extract all the relevant files from the disk image/copy of the source device created in the previous step. This process is done by mounting the disk image in read-only mode and then walking the file system tree. Each file in the entire filesystem is opened in read-only mode to check if their magic numbers (file signature) match a certain criterion. The current version of SynFO provides support to find pictures, audio, video, and archive files.

File in-file embedding, including but not limited to picture files in Word documents, has also been considered for picture files to improve the fidelity of the data extraction phase. SynFO can extract PNG, JPG/JPEG, and GIF picture files by detecting their presence in other files. Table 1 represents all the supported file types from which SynFO can extract picture files, including the method used to extract the files. All the extraction methods are mentioned below in the table.

Table 1. List of supported file formats from different sources.

Source File	Extraction Method
DD	Header/Footer Carving
ISO	Header/Footer Carving
LOG	Header/Footer Carving
PCAP	Header/Footer Carving
PPT	Zip Extraction
PPTX	Zip Extraction
DOC	Zip Extraction
DOCX	Zip Extraction
XLS	Zip Extraction
XLSX	Zip Extraction
ODT	Zip Extraction
ODP	Zip Extraction
ODF	Zip Extraction
PDF	PDF Reading
EPUB	EPUB Reading

3.3. File in File Extraction

Table 1 shows all the methods SynFO employs to extract picture files from other files. This subsection further explains how those extraction methods work. Starting with Header/Footer carving, SynFO uses the classic header/footer file signature matching technique to carve out embedded picture files from other files. This method leverages the fact that every (supported) picture file has a unique set of bytes at the beginning and the end to help identify that file even if it's not indexed anywhere. The same method is employed by many existing file carving software, but a study by [26] dives deeper into file carving. To extract files from office documents like MS Word [27], MS Excel [28], Apple Pages [29], Apple Numbers [30], LibreOffice Writer [31], and others, as mentioned in Table 1, using **Zip Extraction**, tthe algorithm exploits one common feature used by all those document processors, this feature is to use zip archives as their document files. These archived document files contain a common directory containing all the media content. SynFO walks through the directories to find media content to extract all the embedded pictures from the documents. All the embedded picture files are verified using their magic bytes (signature) to avoid false positives. And last but not least, SynFO relies on [32], a third-party open-source library, to enable PDF/EPUB Reading and thus extract picture files from those document files.

3.4. Automated PoI Identification

One of the objectives of the work is to apply an ML technique for the automated identification of persons of interest. SynFO uses dlib and face_recognition libraries that help develop face verification, detection, and recognition related software. The face_recognition library provides a high-level application interface that interacts with the underlying dlib [33] library that was written in C++, this face_recognition [34] library can make use of either Histogram of Gradients or Convolutional Neural Networks to detect faces in an image. After face detection, face landmarks are calculated and presented in a 128-dimension feature space. These landmarks are then used to compare to other faces to verify whether two faces match or not. The script in use here leverages these libraries to provide the following functionalities:

1. One to One Matching
2. One to Many Matching
3. Many to One Matching
4. Many to Many Matching

Provided two datasets of images, one consisting of the collection of images extracted in the previous step (unknown images) and the other one of the POI images (known), the software will try to find the number of faces in each instance. The result could either be 0, 1, or many. In either case, the software will match every face from an unknown set of images with every image of a known set of images. The libraries that are being used currently claim 99.38% accuracy on the LFW dataset. All the poi identification script results are written into a text file for easy referencing and documentation.

4. Presentation

This section presents the utility of SynFO software by explaining how an end user can use the tool on their machine through examples.

SynFO is a command line utility that works on Linux/Unix based machines. This CLI tool keeps the command simple by minimizing the number of options and flags a user needs to use to complete their task. All the flags and options are documented within the CLI tool and can be accessed using the *-h (help)* flag. Following are some command examples:

1. File Extraction

   ```
   synfo_v1 ext -src <path_to_device_file> -dst <path_to_evidence_file> [-ft] [-bs]
   ```

   ext command is used to indicate to SynFO that only extraction of files should be done. All the extracted files are stored in their respective directories by the name of the file type selected; these directories are created in the exact location where the disk image/evidence file is saved

   -src flag expects a device file, a device file is usually found under the directory /dev/ in Linux/Unix based machines

   -dst flag expects the path where the disk image must be saved

   -ft is an optional flag used to specify the type of files that must be extracted; by default, it extracts pictures. Possible options for this flag [image | audio | video | archive]

   -bs is an optional flag that is used to specify the batch size to be used during disk imaging

Command run examples:

```
synfo_v1 ext -src /dev/sdb -dst /home/user1/backup/image1.dd
```

```
synfo_v1 ext -src /dev/sdb -dst /home/user1/backup/image1.dd -ft audio
```

2. Automated PoI Identification:

```
synfo_v1 apd -src <path_to_device_file> -dst <path_to_evidence_file>
-poi <path_to_known_images> [-model] [-bs]
```

apd command is used to indicate SynFO that it must perform Automated PoI identification after extracting all the images from the source. All the extracted images are stored in the images directory, this directory is created in the same location where the disk image/evidence file is saved

-poi expects the path of a directory where a few pictures of the person of interest are stored so they can be matched with the pictures extracted/found from the disk image, extracted images are saved in the images folder, images folder is created in the same location as the disk image

-src flag expects a device file, a device file is usually found under the directory /dev/ in Linux/Unix based machines

-dst flag expects the path where the disk image must be saved

-bs is an optional flag that is used to specify batch size to be used during disk imaging

-model is an optional flag that lets the user select the ML model to detect faces in the pictures. Possible values for this flag are hog or cnn. hog is faster but less accurate, cnn is slower but has higher accuracy

Command run examples

```
synfo_v1 apd -src /dev/sdb -dst /home/user1/backup/image1.dd
```

```
synfo_v1 apd -src /dev/sdb -dst /home/user1/backup/image1.dd -model cnn
```

5. Results and Discussion

This section answers two more important questions:

1. How was SynFO software tested?

2. What were the results post testing?

How was SynFO software tested?
The performance and accuracy of the software was tested with the help of 2 different scenarios with the 16 GB SD Card and 32 GB USB Drive as the evidence. Both the evidence devices were populated with various files based on the scenario, and some of the files were deleted before using the evidence for testing. Experimental results suggest that faces of people can be identified in various scenarios, including pictures where multiple faces are captured in a single image. Pictures with different environmental conditions like street-lamp light at night, rain, day time, night time and pictures that contain faces of different skin color. Pictures from different devices like mobile phone cameras, laptop webcam, and professional cameras were also considered during experimentation.

What were the results post testing?
A snapshot of these experiments is explained below. Table 2 explains the different cameras and

scenarios on which this algorithm was tested. Results are discussed in detail with results of specific experiments, supporting input data, and recorded resultant values through case-based fake scenarios.

Table 2. Performance of SynFO in different scenarios

Scenario	**Camera**	**Multi Face**	**Detection**	**Verification**
Day	Standard DSLR	N	Y	Y
Day	Honor 9N	N	Y	Y
Rain	Standard DSLR	N	Y	Y
Indoors	Standard DSLR	Y	Y	Y
Night	iPhone	N	Y	Y
Night Rain	Honor 9N	Y	Y	N

The image of the person of interest in both the scenarios is kept the same and has been shown below in Figure 1.

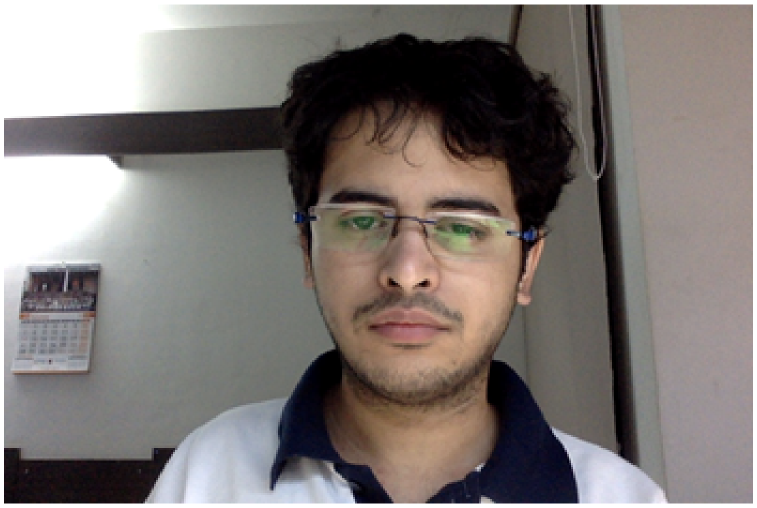


Figure 1. Image of a person of Interest.

Case 1
Apply facial recognition using SynFO by extracting only image files from both the evidence (16 GB and 32 GB). The performance of the proposed utility SynFo and a couple of images matched with the PoI are shown in Table 3, Figure 2, and Figure 3, respectively.

Table 3. Shows time taken for Case 1 (hh:mm:ss)

Device	**Size**	**Imaging Time**	**Extraction Time**	**PoII Time**
SD Card	16 GB	00:22:22	00:01:22	00:00:54
USB Drive	32 GB	00:40:23	00:02:40	00:01:54

Case 2
In the second scenario, the test for SynFO is to detect the image files embedded in containers like docx, pdf, etc. Face recognition was successful in this case also, as we could find images matching the face of the person of interest from the embedded files. The performance of the proposed utility

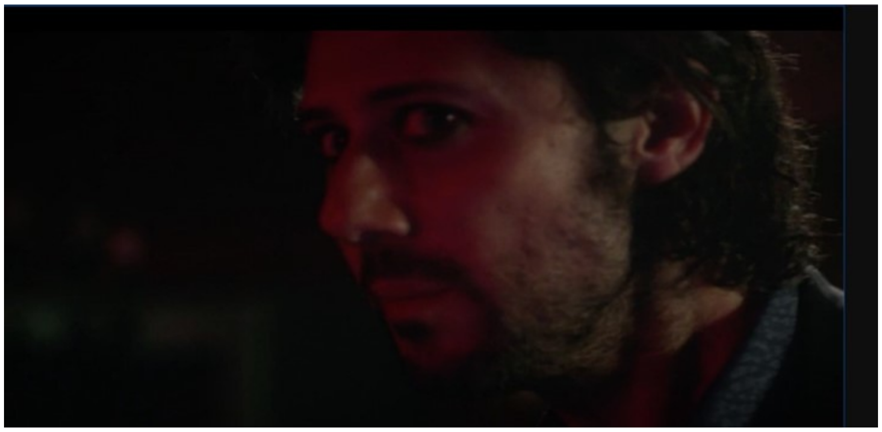


Figure 2. Low quality, bad lighting PoI image.

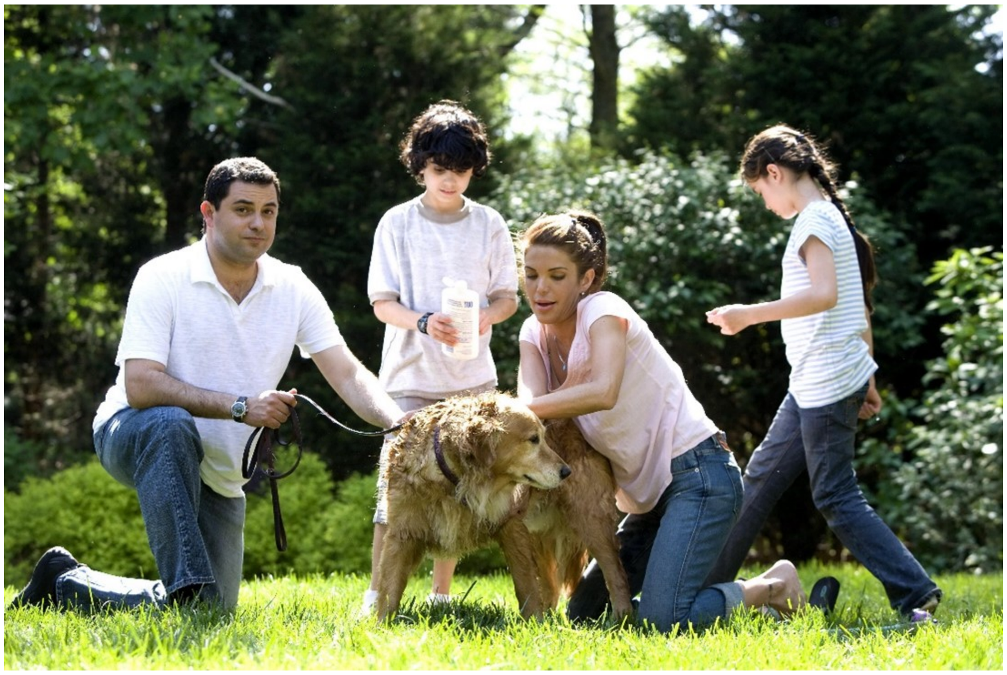
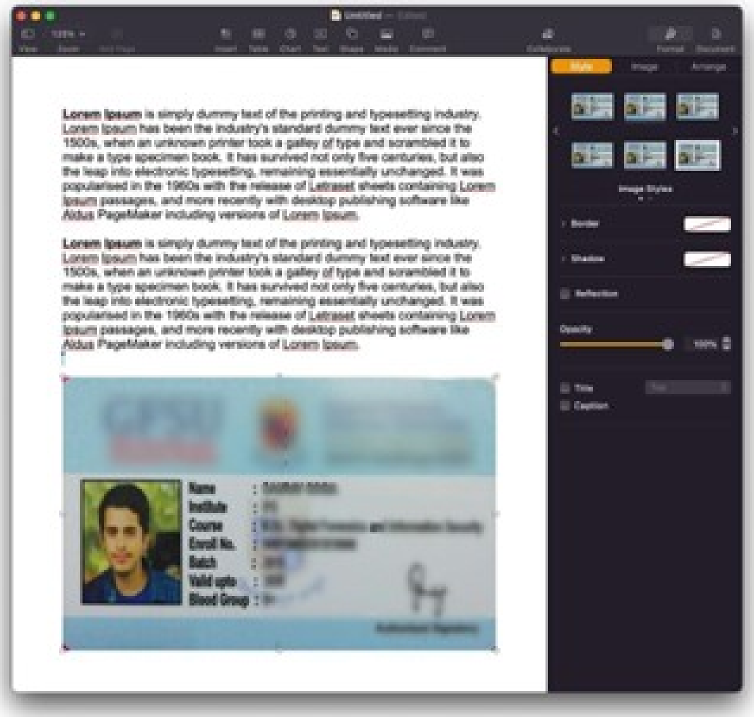


Figure 3. PoI images in different locations with multiple faces in same image.

SynFo, the document containing the image matched with the PoI, and the image itself are shown in Table 4, Figures 4, and 5, respectively.

Table 4. Shows time taken for Case 2 (hh:mm:ss)

Device	Size	Imaging Time	Extraction Time	PoII Time
SD Card	16 GB	00:22:23	00:02:52	00:02:47
USB Drive	32 GB	00:40:39	00:02:56	00:03:03



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Figure 4. Fake Document 1

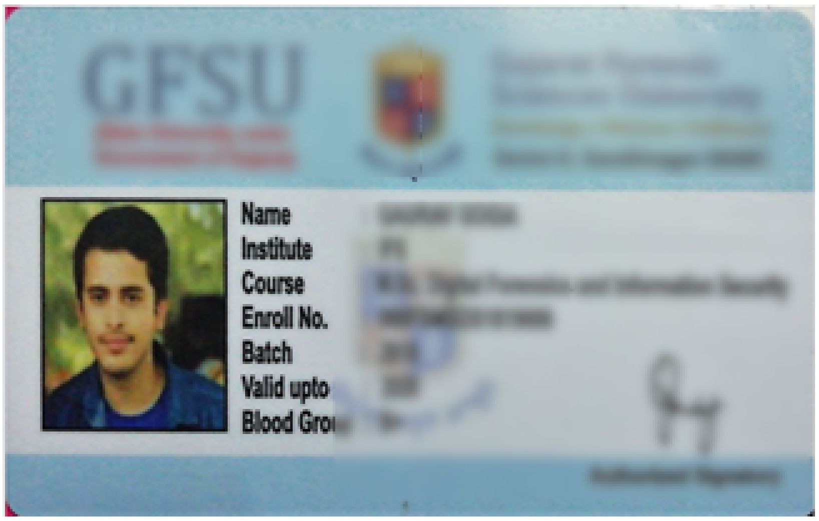


Figure 5. Fake Document 2

6. Limitations and Future Scope

This section further explores the discussion side of our experimental results by diving into the limitations that can be addressed with further research work in this niche. The proposed solution can only process Linux/Unix based operating systems. It has been tested on MacOS Catalina and Arch Linux. Future work will include support for other operating systems like Windows OS. Embedded image extraction may not work with documents created using MS Office 2019 or MS

Office 365 due to the change in how embedded pictures are stored in the newer version documents. Future research work can consider these points and look into the differences between how different versions of MS Office applications save/embed media elements like pictures and videos. Automated image carver only supports PNG, JPG/JPEG, and GIF images; this can be extended by adding header and footer signatures of other image formats.

Further, more extensions can be added by identifying and using header and footer signatures of other file formats, including but not limited to video files and audio files. The automated person of interest verification module uses a library tested on the LFW dataset with over 97% accuracy. However, experimental results explain that in some real-world use cases where pictures of people with and without facial hair, severe light conditions or heavy rain may lead to false positives. Future research work with a larger data set and other competing algorithms can help improve automated PoI identification.

7. Conclusion

The outcome of the work discussed in this article is a command-line ML-based utility that automates acquisition, relevant file extraction, face recognition, and basic reporting tasks to improve investigation speed. The results are highly impressive and can change how face recognition is done manually. This will reduce work hours in manually extracting and checking thousands or even lakhs of images and provide a reliable and accurate mechanism for investigators for these redundant and time-consuming tasks. The work can be extended to increase the scope of container files and recognition of objects in addition to faces. The proposed software will help speed up the investigation of crimes at a larger scale and thus help investigators focus on more complex cases.

References

[1] G. Horsman, "Tool testing and reliability issues in the field of digital forensics," *Digital Investigation*, vol. 28, pp. 163–175, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287618303062

[2] J. Xiao, S. Li, and Q. Xu, "Video-based evidence analysis and extraction in digital forensic investigation," *IEEE Access*, vol. 7, pp. 55 432–55 442, 2019.

[3] A. Morgan and M. Coughlan, "Police use of cctv on the rail network," *Trends and Issues in Crime and Criminal Justice*, pp. 1–17, 10 2018.

[4] D. Lillis, B. A. Becker, T. O'Sullivan, and M. Scanlon, "Current challenges and future research areas for digital forensic investigation," *CoRR*, vol. abs/1604.03850, 2016. [Online]. Available: http://arxiv.org/abs/1604.03850

[5] L. Pasquale, Y. Yu, M. Salehie, L. Cavallaro, T. T. Tun, and B. Nuseibeh, "Requirements-driven adaptive digital forensics," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, 2013, pp. 340–341.

[6] F. Marturana and S. Tacconi, "A machine learning-based triage methodology for automated categorization of digital media," *Digital Investigation*, vol. 10, no. 2, pp. 193–204, 2013, triage in Digital Forensics. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287613000029

[7] M. Al Fahdi, N. Clarke, F. Li, and S. Furnell, "A suspect-oriented intelligent and automated computer forensic analysis," *Digital Investigation*, vol. 18, pp. 65–76, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287616300792

[8] H. Mohammed, N. Clarke, and F. Li, "An automated approach for digital forensic analysis of heterogeneous big data," *Journal of Digital Forensics, Security and Law*, vol. 11, 2016. [Online]. Available: https://commons.erau.edu/jdfsl/vol11/iss2/9

[9] N. Kumar, P. K. Keserwani, and S. G. Samaddar, "A comparative study of machine learning methods for generation of digital forensic validated data," in *2017 Ninth International Conference on Advanced Computing (ICoAC)*, 2017, pp. 15–20.

[10] A. K., S. Grzonkowski, and N. A. Lekhac, "Enabling trust in deep learning models: A digital forensics case study," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 1250–1255.

[11] M. K. Johnson and H. Farid, "Exposing digital forgeries through specular highlights on the eye," in *Information Hiding*, T. Furon, F. Cayre, G. Doërr, and P. Bas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 311–325.

[12] A. Makrushin., T. Neubert., and J. Dittmann., "Automatic generation and detection of visually faultless facial morphs," in *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 6: VISAPP, (VISIGRAPP 2017)*, INSTICC. SciTePress, 2017, pp. 39–50.

[13] T. Neubert, "Face morphing detection: An approach based on image degradation analysis," in *Digital Forensics and Watermarking*, C. Kraetzer, Y.-Q. Shi, J. Dittmann, and H. J. Kim, Eds. Cham: Springer International Publishing, 2017, pp. 93–106.

[14] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2016, pp. 1–6.

[15] T. Huang and X. Yuan, "Detection and classification of various image operations using deep learning technology," in *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 1, 2018, pp. 50–55.

[16] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification based machine learning approach with high order statistics features," in *2016 24th European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1183–1187.

[17] V. Chatzis, F. Panagiotopoulos, and V. Mardiris, "Face to iris area ratio as a feature for children detection in digital forensics applications," in *2016 Digital Media Industry & Academic Forum (DMIAF)*, 2016, pp. 121–124.

[18] S. Tariq, S. Lee, H. Kim, Y. Shin, and S. S. Woo, "Detecting both machine and human created fake face images in the wild," in *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security*, ser. MPS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 81–87. [Online]. Available: https://doi.org/10.1145/3267357.3267367

[19] M. F. E. M. Senan, S. N. H. S. Abdullah, W. M. Kharudin, and N. A. M. Saupi, "Cctv quality assessment for forensics facial recognition analysis," in *2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence*, 2017, pp. 649–655.

[20] G. Amato, F. Falchi, C. Gennaro, F. V. Massoli, N. Passalis, A. Tefas, A. Trivilini, and C. Vairo, "Face verification and recognition for digital forensics and information security," in *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, 2019, pp. 1–6.

[21] A. Bharati, R. Singh, M. Vatsa, and K. W. Bowyer, "Detecting facial retouching using supervised deep learning," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 1903–1913, 2016.

[22] "The go programming language," https://golang.org/, accessed: 2021-08-25.

[23] K. Clark and F. McCabe, "Go! — a multi-paradigm programming language for implementing multi-threaded agents," *Annals of Mathematics and Artificial Intelligence*, vol. 41, p. 171–206, 2004.

[24] "Welcome to python.org," https://www.python.org/, accessed: 2021-08-25.

[25] A. Nagpal and G. Gabrani, "Python for data analytics, scientific and technical applications," in *2019 Amity International Conference on Artificial Intelligence (AICAI)*, 2019, pp. 140–145.

[26] E. Uzun and H. T. Sencar, "Jpg *scraper* : An advanced carver for jpeg files," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1846–1857, 2020.

[27] "Transform your word docs with microsoft 365 | microsoft word," https://www.microsoft.com/en-in/microsoft-365/word, accessed: 2022-03-23.

[28] "Download excel—buy spreadsheet software | microsoft excel," https://www.microsoft.com/en-in/microsoft-365/excel, accessed: 2022-03-23.

[29] "Pages," https://www.apple.com/in/pages/, accessed: 2022-03-23.

[30] "Numbers," https://www.apple.com/in/numbers/, accessed: 2022-03-23.

[31] "Writer | libreoffice—free office suite—based on openoffice—compatible with microsoft," https://www.libreoffice.org/discover/writer/, accessed: 2022-03-23.

[32] "Unidoc," https://github.com/unidoc/unidoc, accessed: 2022-03-23.

[33] "Dlib c++ library," http://dlib.net/, accessed: 2022-03-23.

[34] A. Geitgey, "Face-recognition: Recognize faces from python or from the command line [python]," https://github.com/ageitgey/face_recognition.