A Weighted Rule-Based Model for File Forgery Detection: UA.PT Bioinformatics at ImageCLEF 2019

João Rafael Almeida[1,2][0000−0003−0729−2264], Pedro Freire[1][0000−0001−5663−3403], Olga Fajarda[1][0000−0003−1957−4947], and José Luís Oliveira[1][0000−0002−6672−6176]

[1] DETI / IEETA, University of Aveiro, Aveiro, Portugal
{joao.rafael.almeida, pedrofreire, olga.oliveira, jlo}@ua.pt
[2] Department of Information and Communications Technologies, University of A Coruña, A Coruña, Spain

Abstract. With today's digital technology, disparate kinds of data can be easily manipulated. The forgery commonly hides information, by altering files' extensions, files' signatures, or by using steganography. Consequently, digital forensic examiners are faced with new problems in the detection of these forged files. The lack of automatised approaches to discover these infractions encourages researchers to explore new computational solutions that can help its identification. This paper describes the methodologies used in the ImageCLEFsecurity 2019 challenge, which were mainly rule-based models. The rules and all of their underlying mechanisms created for each task are described. For the third task, was used a random forest algorithm due to the poor performance of these rules.

Keywords: ImageCLEF · File Forgery Detection · Rule-based models

1 Introduction

The ImageCLEF [7] initiative launched a new security challenge, called ImageCLEFsecurity, addressing the problem of automatically identifying forged files and stego images [10]. This challenge is divided into three sub-tasks: 1) *forged file discovery*, 2) *stego image discovery* and 3) *secret message discovery*. The *forged files discovery* sub-task is the first task of the challenge and it is independent from the remaining two tasks. The goal of this task is to automatically detect files whose extension and signature has been altered; more specifically, to identify the files with extension PDF that are, actually, image files (with extension JPG, PNG, and GIF). The objective of the second sub-task is to identify the images that hide steganographic content and the goal of the third task is to retrieve these hidden messages.

In this paper, we present the several approaches that we used to address this challenge. The main solution is based on an orchestration of specialised rule-based models. For each model, a set of rules was defined with the purpose of identifying a specific file or message. Additionally, when there are insufficient rules to provide a good result, other complementary strategies have been combined, namely a random forest classifier.

2 Materials and Methods

For each task, a training set and a test set were provided. The training set of the first task is composed of 2400 files, 1200 of which are PDF files. The remaining files, despite having PDF extension, belong to one of three classes: JPEG, PNG, GIF, each with 400 files. In the second and third tasks, the training sets include 1000 JPEG images, 500 of which are stego images and the others are clean images. In the case of the third task, the stego images contain five different text messages. Regarding the test sets, the first task is comprised of 1000 files and the second and third tasks are composed of 500 images.

In this section, we present the five methods that were used to solve each task of the challenge.

2.1 Rule-Based Approach

A typical rule-based system is constructed through a set of if-then rules [14] which help identify conditions and patterns in the problem domain. However, the use of simple conditions may not be enough to obtain the best results. Sometimes, to accomplish a more accurate outcome, those rules need to be balanced, with weights. The subject of rule weights in fuzzy rule-based models for classifications is not new, and its positive effect has already been proven [8].

We propose a rule-based weighted system with a set of models (figure 1), which are specialised in classifying a specific entry. Each model generates a confidence score regarding the match of the received input with its conditions. The orchestrator collects all the results and chooses the model that gives the best score. When more than one models give similar good confidence scores for different classes, the weights of the rules are readjusted and a new classification cycle is performed to help separating the classes' scores. These readjustments will, hopefully, allow the right output to stand out.

The rules and the weight of the rules are specific to each problem and scenario. Therefore, we used this approach as our base method for all the tasks. The rules and the methods to classify the rules are specified in section 3.

2.2 Image Distortion Pattern Matching

Steganographic techniques permit hiding, within an image, information that should be perceptually and statistically undetectable [2,11]. However, some of these techniques, may not respect these two principles entirely, namely tools like

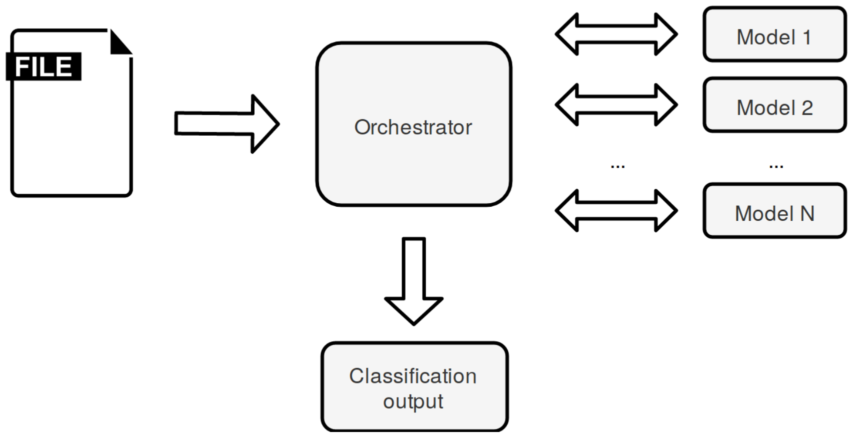

Fig. 1: Proposed rule-based system.

Jsteg, Outguess, F5, JPHide, and Steghide. These tools use the least significant bit (LSB) insertion technique and distort the fidelity of the cover image by choosing the quantized DCT coefficients as their concealment locations [11].

Our approach aims to identify flaws of the used method by searching for a common pattern among all the stego images.

While scanning the training set of the second task for common patterns, it was possible to identify that several stego images had a distortion pattern of 8x8 pixels size, that could easily be identified with the naked eye, as described in figure 2 and figure 3.

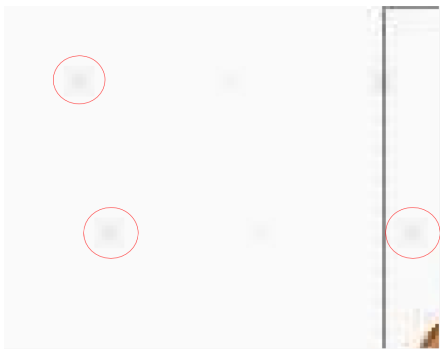

Fig. 2: Distortion patterns observed on a training image of task 2 (with a zoom of 670%).

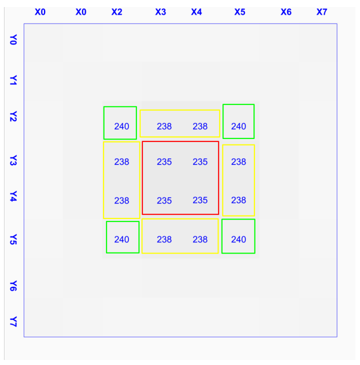

Fig. 3: Pixel block of 8x8 with distortion pattern.

Taking figure 3 as reference, the identified pattern could be described by the following relation between each pixel, where P(x, y) represents the mean value of R, G, and B at position (x,y):

$$\begin{aligned}
P(x3,y3) &= P(x3,y4) = P(x4,y3) = P(x4,y4)\\
P(x2,y3) &= P(x2,y4) = P(x5,y3) = P(x5,y4)\\
&= P(x3,y2) = P(x4,y2) = P(x3,y5) = P(x4,y5)\\
P(x2,y2) &= P(x2,y5) = P(x5,y2) = P(x5,y5)\\
P(x3,y3) &> P(x3,y2)\\
P(x3,y2) &> P(x2,y2)
\end{aligned}$$

We created a function to scan an image for this pattern and to count the number of occurrences. The function determines that a certain image had a message if the number of patterns found is greater than the specified threshold. Its output was used as a parameter into our weighted rule-base model.

2.3 Image Metadata Pattern Searcher

Another approach used to assist the creation of rules for our rule-based model was a pre-analysis of the file's metadata. This analysis aimed to discover patterns that could be used as rules in the model. For instance, in the JPEG images of the training set of the second task, a set of bits were detected which identified the images with a hidden message.

The pattern search was mainly done with the metadata, ignoring the image bitstream. The rational was that the altered files could be signed in the metadata to quickly identify which files are of interest. This simple signature would go unnoticed and it would increase the decoding procedure.

2.4 Random Forests for Rule Definition

Random forest is a supervised learning algorithm developed by Breiman [3], who was influenced by the work of Amit and Geman [1]. This algorithm constructs a large number of decision trees, considering a random subset of features for each split, and makes a prediction by combining the predictions of the different decision trees. Caruana and Niculescu-Mizil [4] performed an empirical comparison of supervised learning and concluded that random forest was one of the algorithms that gave the best average performance.

The random forest algorithm has several positive characteristics [5] for this challenge, namely it can be used for high-dimensional problems and it gives an estimation of the importance of variables. Moreover, it just needs two parameters: the number of trees in the forest (*ntree*) and the number of features in the random subset used in each split (*mtry*).

We used the random forest algorithm in order to help solve the third task and our implementation is described in section 3.

2.5 Manual Tuning

In large data sets, manual classification is unrealistic. However, since the training and test set are small, we decide to try a manual validation. This approach consists mainly in a verification of the rule-based output, followed by manual adjustments considered relevant. This method was, essentially, used in the second task.

When analysing the training set, the 8x8 pixel distortion pattern described in 2.2 was identified. Using the rule-based model in the early stages, made it possible to define rules to reach a precision of 1. However, the recall was low. Therefore, we isolated the images not detected as forged and tried to identify these distortions in the image manually. This procedure increased our recall significantly.

3 Results

The described methods were combined and led to several submissions in the different tasks. The performance of the submissions was evaluated using the traditional metrics: precision, recall, and F1, in the first two tasks and edit distance in the third task.

In the first task, the precision was defined as the quotient between the number of altered images correctly detected and the total number of files identified as changed. In its turn, the recall was the quotient between the number of altered images correctly detected and the total number of files modified.

For the second task, the definition of precision and recall was similar. The precision was the quotient between the number of images with hidden messages correctly detected and the total number of images with hidden messages identified. The recall was the quotient between the number of images with hidden messages correctly detected and the total number of images with hidden messages.

Finally, the third task used the edit distance to measure the efficiency into recovering the message. This distance is the minimum-weight count of edit operations that transforms a string in another one.

3.1 Task 1: Identify Forged Images

Detecting the type of file is a process that can be done using three different file characteristics: the file's extension, the magic number, and the file's content [9]. The most straightforward technique to hide a file is changing the file's extension and the magic number, which is a set of standard bytes that signs the file. With this technique, the operation system is unable to open the file. Therefore, four models are built, each one specialised in identifying a file type (PDF, PNG, JPG or GIF). Each model produces a score reflecting the confidence that the analysed file is of the given type. These scores are sent to the orchestrator, who classifies the type of file based on the scores received.

The initial approach considered standard flags in the file structure, such as the last bytes or the number of occurrences of a set of bytes. For instance, a JPEG file has the hexadecimal 0xFFDA at least once in its structure because this is the flag that indicates where the image binary starts. Table 1 presents the flags for the end of file for each file type. For this first task, we used the rule of identifying the end of file flag and obtained an F1 measure of 1.0.

Table 1: Rule used in task one.

File format	**Ending of file (Hexadecimal values)**
GIF	0x3B
JPG	0xFFD9
PNG	0x49454E44AE426082
PDF	0x454F46

3.2 Task 2: Identify Stego Images

In JPEG images there are two different stages of compression: lossy and lossless. Embedding steganographic content inside images that uses lossy compression increases the possibility of that content to be partly lost, which means that, it is not feasible to hide a message on a lossy stage. Therefore, in the case of JPEG files, the steganography should take place on the lossless stage. The DCT and the quantization phase form part of the lossy stage, while the Huffman encoding used to further compress the data is lossless. [13].

After scanning the training set of the second task for common parts that distinguishes stego images binary data from non-stego images, it was possible to observe some patterns and specify the weights of the rules that identifies stego images. The patterns and their weights are described on table 2. We could observe that all the patterns appeared in the Huffman table sequences, i.e. after the marker DHT. The sequences in a JPEG file are identified by a two-byte code described in table 3.

Table 2: Weighted rules used in Task 2 for the first submission.

Rule	**Searched on**	**Hex matching value**	**Weight**
1	DHT	0x3435363738393a434445464748494a53545556 5758595a636465666768696a737475767778797a	0.35
2	DHT	0x35363738393a434445464748494a535455565 758595a636465666768696a737475767778797a	0.35
3	DHT	0xf2f3f4f5f6f7f8f9fa	0.30

Table 3: Common JPEG file markers [6].

Marker id	**Short value**	**Description**
SOI	0xFFD8	start of image
APPn	0xFFEn	application data
DQT	0xFFDB	quant.tables
DHT	0xFFC4	Huffmantables
SOF	0xFFCn	start of frame
SOS	0xFFDA	start of scan
DRI	0xFFDD	restart interval
RSTn	0xFFDn	restart
COM	0xFFFE	comment
EOI	0xFFD9	end of image

This task was also solved using the rule based-model where the images with a score equal to or greater than 0.70 were considered as altered. However, in this case, strategies and different ways to extract information from the images were combined. Initially, a metadata pattern searcher to compare the metadata fields' content was developed. From this analysis, in all the images with a hidden message, the set of bits represented in the rules displayed in table 2 were found. However, these rules produced several false positives, achieving, using the training set, a recall of 1.0 and a precision of 0.75.

Due to the lack of precision, we attempt to identify the distortion pattern in the images, the method described in section 2.2. Without the rules used in the first approach and using a threshold of at least one pixel-block with the distortion, this method produced a precision of 0.53 and a recall of 0.60 in the training set. It was also the best score obtained from all runs using this approach isolated.

Then, to increase the precision, the decision was made to combine the rules of the first approach with this analyser. The image analyser method was only used when an image was classified through the first approach as having a hidden message. This decreased the recall to 0.604 and the precision remained in the 0.75, the best precision result so far. The decrease of the recall and the bad results in isolated scenarios led to the abandonment of this approach and to focusing only on ways to increase the quality of the rules.

At this stage, a submission was made, obtaining an F1 measure of 0.933 and a precision of 0.874.

In the next attempts some manual rectifications were made in the output retrieved from the rule-based system, by observing the images classified as having a message. Some submission were made following only the rule-based approach mixed with the manual validation, and the best result was 0.98, both for F1 and precision.

As a last attempt, the decision was made to re-run the metadata pattern searcher to be more precise. Now, it analysed all the metadata as a binary, ignoring which were the fields or its content. With these changes, the method found a new pattern, which produced a new rule, represented in table 4, made it possible to achieve F1 measure of 1 in the training and test sets.

Table 4: Final weighted rules used in Task 2.

Rule	**Searched on**	**Hex matching value**	**Weight**
1	DHT	0x3435363738393a434445464748494a53545556 5758595a636465666768696a737475767778797a	0.35
2	DHT	0x35363738393a434445464748494a535455565 758595a636465666768696a737475767778797a	0.15
3	DHT	0xf2f3f4f5f6f7f8f9fa	0.15
4	APP0	0x00600060	0.35

3.3 Task 3: Retrieve the Message

The goal of the third task was to retrieve the hidden text messages from the stego images. To address this task, initially freely available steganographic tools were used, specifically, Hide'N'Send, Image Steganography [3], QuickStego, SSuite Picsel [4], Steghide [5], and SteganPEG [15]. However, none of the steganographic tools were able to retrieve the hidden text messages.

In our second approach, the RGB matrix was analyzed. First, each colour component was individually inspected, examining the least and the two least significant bits, in order to detect if they could compose ASCII codes of letters in the alphabet, more precisely, the ASCII character in the range from 65 to 90, and from 97 to 122. As these procedures did not provide a pattern for the stored messages, the decision was made to look to the pixel as a whole, inspecting the three colour component combined. We, also, tried to use the two least significant bits from the four pixels that are in the centre of each 8x8 pixel block.

The second approach could not retrieve the hidden messages from the image files and therefore an attempt to find a pattern using the DCT matrix was made, by inspecting the least and the two least significant bits from the value in the first cell of an 8x8 block. The change of the least or the two least significant bit of these values would create a small change in the block brightness, which would explain the distortion identified in the 8x8 pixel block.

[3] https://incoherency.co.uk/image-steganography/
[4] https://www.ssuiteoffice.com/software/ssuitepicselsecurity.htm
[5] http://steghide.sourceforge.net/

None of the procedures described so far could find a pattern in the images of the training set with the same hidden text message. Therefore, the random forest algorithm in an attempt to find a pattern in the binary of the image files was used. The 500 stego images of the training set have, as hidden message, one of five messages. Consequently, the next step was to consider a multiclass classification problem which consists in classifying each stego image into one of the five messages. Initially, for our first model, we used as features the frequency, in percentage, with which each ASCII character appears in the binary of the image files. For the second model, in addition to the features used in the first model, the percentage of 0s and 1s in the binary of the image files were used. To train the models we used the R package *caret* [12] and used cross-validation the chose the optimal value of the parameters *ntree* and *mtry*. In what concerns the performance of the models, this was evaluated using 10-fold cross-validation. Table 5 presents the parameters used and the accuracy of each model.

Table 5: Random forest models' parameters and results.

Models	**Parameters**		**Accuracy**
	ntree	**mtry**	
First model	247	82	31,57%
Second model	374	59	31,33%

From the results, the random forest models were unable to find a pattern in the binary of the image files. In the absence of alternatives, the two random forest models to classify the image files of the test set were used, despite the fact that this task was not a classification problem. The first step was to use the rule based-model defined in the second task to identify the stego images of the test set and, subsequently, the random forest models to classify the images identified as containing a hidden message. For the submissions, all the images in the test set should have a string appointed and therefore an empty string to the images identified as having no hidden message was assigned. Using the first model, an edit distance of 0.588 was obtained and using the second model an edit distance of 0.587. Our best edit distance (0.598) was achieved by assigning the string "*name John Fraud*" to all images we identified as stego images and an empty string to the other images of the test set. These results reflect the fact that we could correctly identify the images with no hidden messages.

4 Conclusion

This paper presents the methodologies to identify forged files and stenographic images used in the ImageCLEFsecurity challenge. These methods were developed specifically for the tasks of this challenge, which does not invalidate them being used for other data sets. In the first task, an F1 measure of 1 was obtained. This excellent result was accomplished mainly because the changes done to the files were only the traditional ones, and with simple rules, it was possible to identify each type. The second task also had a submission with F1 measure of 1. In this case, we could identify a signature in the altered images. On the other hand, in the third task, the best submission had an edit distance of 0.598, mainly due to the success of identifying empty strings, i.e., images without a message. The purposed methodology works if it is possible to define the right rules. The problem in this task was the difficulty to find the stenographic algorithm used.

This challenge allowed for the identification of problems in the developed approach, and most importantly, ways to improve some of these issues. A future work originated from this year's participation could be the creation of a rule generator to fed the rule-based models. The message identification task may be improved by creating a database of strategies used by stenographic attackers, mixed with machine learning approach that look into neighbourhood pixel colour.

Acknowledgements

This work was supported by the projects NETDIAMOND (POCI-01-0145-FEDER-016385) and SOCA (CENTRO-01-0145-FEDER-000010), co-funded by Centro 2020 program, Portugal 2020, European Union.

References

1. Amit, Y., Geman, D.: Shape quantization and recognition with randomized trees. Neural computation **9**(7), 1545–1588 (1997)
2. Attaby, A.A., Ahmed, M.F.M., Alsammak, A.K.: Data hiding inside jpeg images with high resistance to steganalysis using a novel technique: Dct-m3. Ain Shams Engineering Journal (2017)
3. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
4. Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In: Proceedings of the 23rd international conference on Machine learning. pp. 161–168. ACM (2006)
5. Cutler, A., Cutler, D.R., Stevens, J.R.: Random forests. In: Ensemble machine learning, pp. 157–175. Springer (2012)
6. Gloe, T.: Forensic analysis of ordered data structures on the example of jpeg files. In: 2012 IEEE International Workshop on Information Forensics and Security (WIFS). pp. 139–144. IEEE (2012)
7. Ionescu, B., Müller, H., Péteri, R., Cid, Y.D., Liauchuk, V., Kovalev, V., Klimuk, D., Tarasau, A., Abacha, A.B., Hasan, S.A., Datla, V., Liu, J., Demner-Fushman, D., Dang-Nguyen, D.T., Piras, L., Riegler, M., Tran, M.T., Lux, M., Gurrin, C., Pelka, O., Friedrich, C.M., de Herrera, A.G.S., Garcia, N., Kavallieratou, E., del Blanco, C.R., Rodríguez, C.C., Vasillopoulos, N., Karampidis, K., Chamberlain, J., Clark, A., Campello, A.: ImageCLEF 2019: Multimedia retrieval in medicine, lifelogging, security and nature. In: Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the 10th International Conference of

the CLEF Association (CLEF 2019), LNCS Lecture Notes in Computer Science, Springer, Lugano, Switzerland (September 9-12 2019)
8. Ishibuchi, H., Nakashima, T.: Effect of rule weights in fuzzy rule-based classification systems. IEEE Transactions on Fuzzy Systems **9**(4), 506–515 (2001)
9. Karampidis, K., Papadourakis, G.: File type identification for digital forensics. In: International Conference on Advanced Information Systems Engineering. pp. 266–274. Springer (2016)
10. Karampidis, K., Vasillopoulos, N., Cuevas Rodrguez, C., del Blanco, C.R., Kavallieratou, E., Garcia, N.: Overview of the ImageCLEFsecurity 2019 task. In: CLEF2019 Working Notes. CEUR Workshop Proceedings (CEUR-WS.org), ISSN 1613-0073, http://ceur-ws.org/Vol-2380/ (2019)
11. Khalid, S.K.A., Deris, M.M., Mohamad, K.M.: A steganographic technique for highly compressed jpeg images. In: The Second International Conference on Informatics Engineering & Information Science (ICIEIS2013). pp. 107–118 (2013)
12. Kuhn, M., et al.: Building predictive models in r using the caret package. Journal of statistical software **28**(5), 1–26 (2008)
13. Kumari, M., Khare, A., Khare, P.: Jpeg compression steganography & crypography using image-adaptation technique. journal of advances in information technology **1**(3), 141–145 (2010)
14. Liu, H., Gegov, A., Cocea, M.: Rule-based systems: a granular computing perspective. Granular Computing **1**(4), 259–274 (2016)
15. Reddy, V.L., Subramanyam, A., Reddy, P.C.: Steganpeg steganography+ jpeg. In: 2011 International Conference on Ubiquitous Computing and Multimedia Applications. pp. 42–48. IEEE (2011)