Detection of Duplicated Image Regions using Cellular Automata

Dijana Tralic[1], Paul L. Rosin[2], Xianfang Sun[2], Sonja Grgic[1]
[1]Faculty of Electrical Engineering and Computing, University of Zagreb,
Unska 3, 10000 Zagreb, Croatia
[2]School of Computer Science and Informatics, Cardiff University
Queens Buildings, 5 The Parade, Cardiff CF24 3AA, UK
Dijana.Tralic@fer.hr

***Abstract*—A common image forgery method is copy-move forgery (CMF), where part of an image is copied and moved to a new location. Identification of CMF can be conducted by detection of duplicated regions in the image. This paper presents a new approach for CMF detection where cellular automata (CA) are used. The main idea is to divide an image into overlapping blocks and use CA to learn a set of rules. Those rules appropriately describe the intensity changes in every block and are used as features for detection of duplicated areas in the image. Use of CA for image processing implies use of pixels' intensities as cell states, leading to a combinatorial explosion in the number of possible rules and subsets of those rules. Therefore, we propose a reduced description based on a proper binary representation using local binary patterns (LBPs). For detection of plain CMF, where no transformation of the copied area is applied, sufficient detection is accomplished by 1D CA. The main issue of the proposed method is its sensitivity to post-processing methods, such as the addition of noise or blurring. Coping with that is possible by pre-processing of the image using an averaging filter.**

***Index Terms*—Image Forensics; Copy-Move Forgery; Cellular Automata; Post-processing**

I. Introduction

Thanks to the simplicity of their handling and the development of a wide range of sophisticated processing tools, digital images completely surplanted the use of analogue images. However, the development of processing tools also allowed modification of digital image content to be done without any visible traces [1]. There are numerous different methods of image forgery, but the most common type is copy-move forgery (CMF) [2] which produces duplicated regions on the same image. It is performed by selecting a part of the image and coping it to a new location in that image with the aim of hiding or adding some objects or content. CMF is very often used, not only because of the simplicity of its application, but thanks to the fact that all the properties of duplicated regions are very well matched with the rest of the image. Additionally some post-processing methods can be applied after forgery with the goal to better fit the forged region in the image or to make detection using local properties impossible.

The aim of digital image forensics is to cope with any kind of image modifications and to determine the authenticity of digital images. Detection of CMF can be done using different approaches, but the most basic is based on dividing the image into overlapping blocks and determining similarity between them by comparing the values of feature vectors. Many different properties can be used as feature sets, such as DCT coefficients [3] or pixel intensity values [4], [5], but there is no unique technique for detection of all manipulation.

A new passive, block-based approach for CMF detection is presented in this paper. The main idea is to divide an image into overlapping blocks and to calculate feature vectors for each block using cellular automata (CA), because it can properly describe texture of blocks by learning a set of rules for those blocks. Rules are then used as a similarity criterion to distinguish duplicated blocks. An issue that arises is that large number of pixel intensities in greyscale images results in a combinatorial explosion in the number of possible rules and an even larger number of possible subsets of rules. A compact description is accomplished by binary representation of the image using local binary patterns (LBPs) [6].

This paper is organized as follows. In Section II CMF and block-based detection is described. Section III presents a new method for CMF detection based on cellular automata. Testing results can be found in Section IV. Future work and conclusion is given in Section V.

II. Copy-move forgery

The type of image forgery where part of an image is selected, copied and moved to a new location in the same image is called copy-move forgery (CMF) [2]. The result of CMF is usually the detection of two duplicated regions in the image because the purpose is to hide or to add some content or object in the image. Many properties such as camera noise or illumination conditions are very well matched in this case so they cannot be used for forgery detection. CMF is also very simple to apply due to the fact that the copied region can be easily fitted to the rest of the image. Examples of CMF from the CoMoFoD database are given in Fig. 2 and 3.

The simplest type of CMF does not include any transformation of the copied area. More complex CMF types are accomplished by rotation, scaling, distortion or applying more than one transformation of the copied area [7]. Additionally some post-processing (addition of noise, image blurring, JPEG compression, etc.) of the forged image can be applied to better hide forgery traces or to deceive detection algorithm.

A. Block-based detection method

CMF detection is possible in many different ways, but the basic method is to divide the image into small, overlapping blocks. For every block a set of features is calculated, and used as a criterion for detection of similar blocks (Fig. 1):

1) pre-processing – this step usually includes just converting of images to greyscale space.
2) blocking – an image is divided into overlapping blocks of size $b \times b$. Due to the fact that sliding by one pixel is used, dividing an $N \times M$ image into $b \times b$ blocks produces $(N - b + 1) \times (M - b + 1)$ blocks.
3) calculation of feature vectors f – some properties of blocks are used to define a proper description of every block. Different feature vectors have been proposed such as Discrete Cosine Transform (DCT) [3], Discrete Wavelet Transform (DWT) [8], Principal Component Analysis (PCA) [9], Zernike moments [10], etc.
4) sorting – grouping of similar blocks is accomplished by some sorting method (e.g. lexicography sorting).
5) detection of similar blocks – feature vectors are compared by calculating the Euclidean distance, possible results are selected according to equation (1). Threshold T_s is usually experimentally determined.

$$\sqrt{\sum_{i=1}^{size(f)} (f_1(i) - f_2(i))^2 \leq T_s} \tag{1}$$

6) discarding neighbourhood blocks – for every pair in the set of possible results the Euclidean distance is calculated using the coordinates of blocks and some pairs are removed (2). Threshold T_d is usually defined according to the selected block size.

$$\sqrt{(x_{f1} - x_{f2})^2 + (y_{f1} - y_{f2})^2} \leq T_d \tag{2}$$

7) detection results generation – all remaining pairs of blocks are marked as forged. Small, falsely detected areas can be removed by some post-processing method.

III. Proposed method

Cellular automata (CA) are a discrete system that contains a regular grid of cells in one finite-state determined by the previous states of a surrounding neighbourhood of cells [11]. Application of CA for image processing is especially interesting because very simple rules can result in complex behaviour. The proposed method is based on the assumption that similar areas in the image should have similar sets of rules produced by the same CA. That property is used to define feature vectors in our approach, which is a variation of block-based methods (it differs only in the feature vectors step as shown in Fig. 1).

A. Representation of image in binary form

The goal in the detection of a plain CMF is simply to find duplicated areas in the image. However, application of CA on a greyscale image requires taking 256 intensity values as cell states, resulting in the combinatorial explosion in the number of possible subsets of rules. To cope with that, we propose representing the image in binary form based on local binary patterns (LBPs) [6], so that only two values are used as a cell states. The LBP of neighbourhood *P* and radius *R* is obtained by thresholding the values of neighbourhood pixels g_p using the value of the central pixel g_c:

$$LPB(P, R) = \sum_{p=0}^{P-1} s(g_p - g_c) \times 2^p, \tag{3}$$

$$s(x) = \begin{cases} 1, & x \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

B. New feature vectors

A new set of feature vectors is defined as a subset of rules from all possible rules selected using CA (Fig. 1):

1) For every pixel p_c define neighbourhood *N* as a group of *k* pixels from the row of the image above pixel p_c. One pixel straight above pixel p_c and an equal number of neighbouring pixels from both sides of that pixel are selected (5).

$$N(p_c) = N(p_{x,y}) = \{p_{x+i,y-1}\}, \quad i = (-\lfloor k/2 \rfloor, ..., \lfloor k/2 \rfloor) \tag{5}$$

2) For the defined neighbourhood calculate a binary representation based on LBP – use the mean value of intensities of the pixel p_c and its neighbourhood for thresholding to binary values b_i (6).

$$b_i = \begin{cases} 1, & p_i \geq mean(N(p_c) \cup p_c) \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

3) Generate a rule for learning the pixel p_c using binary values from the neighbourhood. We use the fast rule identification algorithm proposed by [12] which uses a model whose parameters are estimated based on a minimum variance criterion.

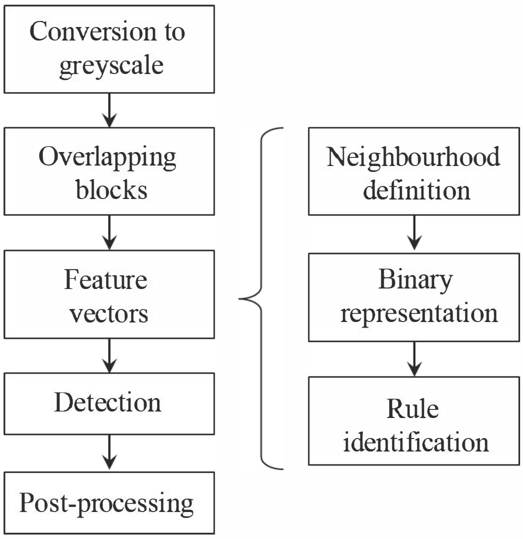


Fig. 1: Main steps of proposed algorithm

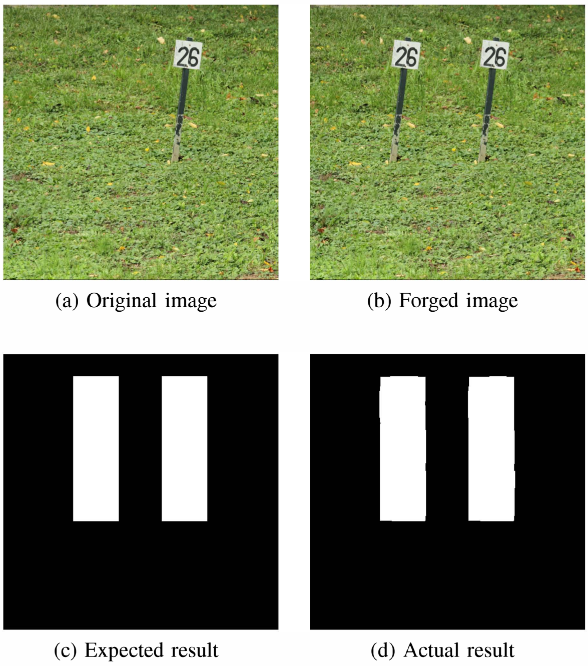
IV. Testing results

Testing is done on 40 images from the CoMoFoD database [7] that contain only translation of the copied area (plain CMF). For all tested cases the following setup is used: 1D CA with neighbourhood of 7 pixels; block size is set to 32 pixels (16 pixels gave better results in some examples); threshold T_d = 40; no removal of falsely detected areas is used; conversion of image in greyscale space is applied prior to detection.

A. Plain CMF

Two examples of plain CMF detection are given in Fig. 2 and 3. For both examples threshold T_s is set to 0. Figure 2 shows an example of very accurate detection on the image which contains complex texture. The copied area is completely detected and there are no falsely detected blocks. However, detection is less accurate in conditions where many areas have similar properties, e.g. homogeneous regions (sky) as shown in Fig. 3. An issue arises in the thresholding process because blocks with very small differences got the same binary representation, and so many blocks are falsely detected.

Results for all 40 images indicated that 80% of tested images have a F-measure score strictly above 0.6, which leads to the conclusion that detection is satisfactory. The presented method is comparable with other methods for CMF detection [3], [9], [10]. Weaker performances are noticeable only in the case when large homogeneous regions are present, in which cases methods that do not need binary representation achieve better results. A possible solution is to find a better representation for a more discriminative description of regions with small differences. On the other hand, the advantage of the proposed method is the very small number of falsely detected regions in all other tested cases.

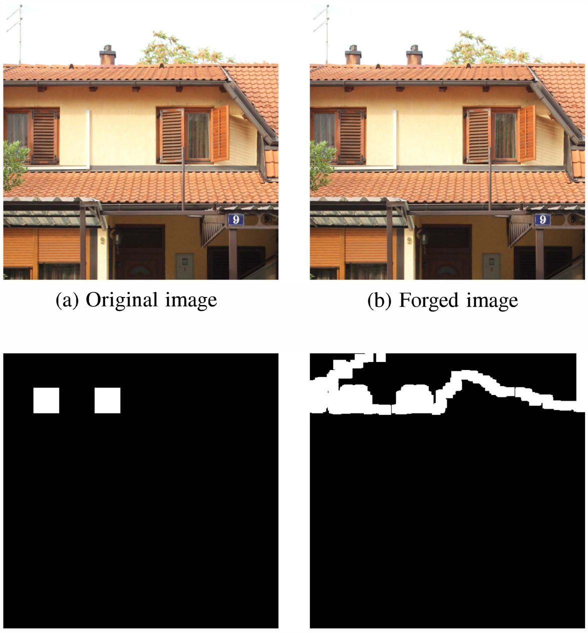

(a) Original image

(b) Forged image

(c) Expected result

(d) Actual result

Fig. 2: Example of successful CMF detection (040_F)


(a) Original image

(b) Forged image

(c) Expected result

(d) Actual result

Fig. 3: Example of unsuccessful CMF detection (003_F)

B. Post-processing methods

The problem in detection of post-processed images is the introduction of differences in pixels values of the copied regions, therefore it is not sufficient to search for two duplicated regions in the image. The most common post-processing methods are addition of noise, JPEG compression and image blurring. Reliable detection is expected in the case of image blurring because blurring is accomplished by calculating the mean value of pixel intensities in a $z \times z$ sliding window (therefore it is omitted from further discussion).

1) Addition of noise: Figure 4 shows an example of detection in an image with added Gaussian noise of zero mean and variance equal to 0.00001 (note that image intensities were normalized to range [0,1] prior to addition of noise). The value of threshold T_s is set to 7. It is possible to see that detection is difficult even when only a small amount of noise is applied. Figure 4a shows that only a small part of the copied area is detected as forged, and there are also some falsely detected blocks. The value of the F-measure is equal to 0.6357. A larger part of the copied area can be detected by selecting higher values for threshold T_s, but that would also result in more falsely detected blocks. Addition of larger amounts of noise would significantly change pixels' values and make detection completely impossible.

Coping with detection of noisy images is however possible by pre-processing of the image with an averaging filter. Figure 4b shows an example of detection after application of an averaging filter of size 3×3. It is visible that accuracy is significantly higher in the case when pre-processing is used. Additionally, the value of the F-measure is 0.8196, pointing to very accurate and robust detection.

2) JPEG compression: Detection of forged images after JPEG compression presents a serious problem for the presented method, as well as for most other forgery detection algorithms. The issue that arises is a result of the compression process where every block of the image is treated separately, leading to larger differences in pixels' values. The consequence is that the same blocks have completely different binary representations, and so different sets of rules are generated. However, like in the case of addition of noise, dealing with JPEG compression is possible by pre-processing the image using an averaging filter.

Figure 5a presents an example of forgery detection on the image with JPEG compression where detection is completely impossible (F-measure = 0.0321). The value of threshold T_s is set to 7. However, a significantly better result is accomplished after filtering of the forged image (after JPEG compression) by an averaging filter of size 3×3 (Fig. 5b). Value of F-measure is equal to 0.3854 (note that no post-processing for removal of falsely detected blocks is used). Furthermore, even better results can be accomplished by more blurring.

V. Conclusion

This paper presents a new method for detection of duplicated regions aimed to cope with CMF. The method is based on dividing the image into overlapping blocks, and calculating some features for every defined block. The set of features is generated by applying cellular automata (CA). However, use of cellular automata for image processing results in an explosion of the number of possible rules, because all pixels' intensities should be used as cell states. To cope with that, we propose a new, compact representation of the image in binary based on local binary patterns. Thank to this binary representation, the number of possible rules is significantly reduced because only two states are used as a cell states.

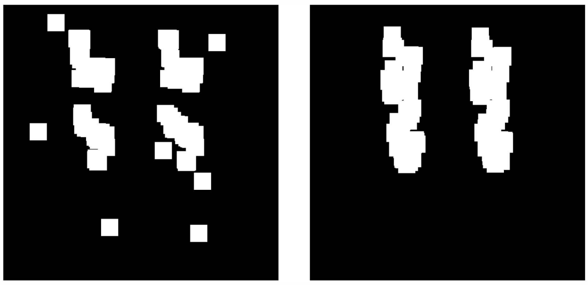

(a) Result without pre-processing (b) Result after pre-processing

Fig. 4: Detection of noisy image (040_F)

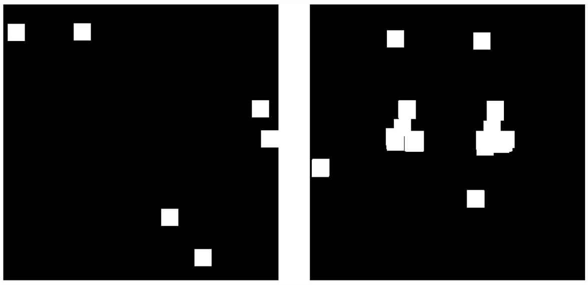

(a) Result without pre-processing (b) Result with pre-processing

Fig. 5: Detection of image with JPEG compression (040_F)

Testing results shows very accurate detection in most cases, with very low false detection rate. However, there are some cases when detection is not satisfactory, such as the presence of areas with many pixels of similar values. In that case, information of texture is lost in the transformation to binary, and many blocks are falsely detected together with duplicated regions. To solve this issue, a better representation of texture should be introduced.

The proposed method is also tested against some post-processing methods. Coping with the addition of noise and JPEG compression is possible only when pre-processing of the forged image is applied. Pre-processing is performed using an averaging filter prior to the detection process.

Future work is aimed at the modification of the proposed method for detection of rotation and scaling of the copied region. Coping with rotation could be achieved by defining a circular neighbourhood and using it to learn rules in a circular way. Detection of scaling could be solved by defining a neighbourhood of different sizes for every block in the image.

References

[1] H. Farid, "Image forgery detection: A survey," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 16–35, 2009.

[2] B. L. Shivakumar and S. Baboo, "Detecting Copy-Move Forgery in Digital Images: A Survey and Analysis of Current Methods," *Global Journal of Computer Science and Technology*, vol. 10, no. 7, pp. 61–65, 2010.

[3] J. Fridrich, D. Soukal, and J. Lukás, "Detection of copy move forgery in digital images," *Proc. Digital Forensic Research Workshop*, 2003.

[4] H. Lin, C. Wang, and Y. Kao, "Fast Copy-Move Forgery Detection," *WSEAS Transactions on Signal Processing*, vol. 5, no. 5, pp. 188–197, 2009.

[5] W. Luo, J. Huang, and G. Qiu, "Robust Detection of Region-Duplication Forgery in Digital Images," *IEEE Information Forensics and Security*, vol. 4, pp. 746–749, 2006.

[6] T. Ojala, M. Pietikainen, and T. Maeenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[7] D. Tralic, I. Zupancic, S. Grgic, and M. Grgic, "CoMoFod-New Database for Copy-Move Forgery Detection," in *Proc. 55th International Symposium ELMAR-2013*, pp. 49–54, 2013.

[8] M. Bashar, K. Noda, N. Ohnishi, and K. Mori, "Exploring Duplicated Regions in Natural Images," *IEEE Transactions on Image Processing*, 2010. accepted for publication.

[9] A. Popescu and H. Farid, "Exposing digital forgeries by detecting duplicated image regions," tech. rep. tr2004-515, Dartmouth College, 2004.

[10] S. Ryu, M. Lee, and H. Lee, "Detection of Copy-Rotate-Move Forgery using Zernike Moments," in *Information Hiding Conference*, pp. 51–65, 2010.

[11] P. L. Rosin, "Training Cellular Automata for Image Processing," *IEEE Transaction on image processing*, vol. 15, no. 7, pp. 2076–20874, 2007.

[12] X. Sun, P. L. Rosin, and R. R. Martin, "Fast Rule Identification and Neighborhood Selection for Cellular Automata," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. 41, no. 3, pp. 749–760, 2011.