Content-aware Detection of JPEG Grid Inconsistencies for Intuitive Image Forensics

Chryssanthi Iakovidou[1], Markos Zampoglou,[1], Symeon Papadopoulos[1] and Yiannis Kompatsiaris[1],

[(1)] *Information Technologies Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece 6th km Harilaou - Thermi, 57001, Thessaloniki*

c.iakovidou@iti.gr, markzampoglou@iti.gr, papadop@iti.gr, ikom@iti.gr.

Abstract

The paper proposes a novel method for detecting indicators of image forgery by locating grid alignment abnormalities in JPEG compressed image bitmaps. The method evaluates multiple grid positions with respect to a fitting function, and areas of lower contribution are identified as grid discontinuities and possibly tampered areas. An image segmentation step is introduced to differentiate between discontinuities produced by tampering and those that are attributed to image content, making the output maps easier to interpret by suppressing non-relevant activations. Our evaluations, on both synthetically produced datasets and real world tampering cases against seven methods from the literature, highlight the effectiveness of the proposed method in its ability to produce output maps that are clear and readable, and which can achieve successful detections on cases where other algorithms fail.

Keywords: Image forensics, JPEG artifacts, forgery localization, splicing

1. Introduction

Digital images have become an integral part of everyday life and, arguably, one of the most popular ways to convey a message. Exploiting the natural human tendency to give priority to visual information, digital images are widely utilized as a means to convince audiences, engage users, augment storytelling, and provide evidence in various domains from business and marketing to journalism and law, to name a few.

Given the proliferation and wide availability of image processing tools, the authenticity of a digital image cannot be taken for granted. A doctored image can influence the opinion of viewers and have serious consequences on peoples' beliefs and attitudes. To this end, there has recently been a growing interest in algorithms to verify the authenticity and integrity of digital images.

Image forgery detection techniques are often categorized into two classes: (i) active methods, which rely on an embedded digital signature that is encoded at the source side (e.g., by the capturing device) and verified at the receiver's end; (ii) passive (blind) methods, that require no prior information but instead base their detection on the assumption that the tampering process may leave invisible but detectable traces on the image.

Even though active methods can be very reliable, their use is not possible in situations where content from unknown or untrusted sources may contain important information [1]. In such cases the assessment of content authenticity is based on what has come to be referred to as *intrinsic fingerprints*, i.e. inherent traces left from various post-processing operations. The type and salience of traces left by tampering depends on multiple factors, such as the type of tampering, the image format and compression parameters.

A recent study on splicing localization[1] [2] pointed out a big discrepancy between real-world cases of tampering and the benchmark datasets that are typically used in academic literature.

Motivated by this finding, in this work we are interested in extending the arsenal of tampering detection tools by proposing a novel method. The method aims to be applicable to a wide range of real-world image forgeries and practical for users with no specialized training in interpreting forensic maps. The method is based on a technique that searches for JPEG blocking artifact discontinuities as a sign of possible forgery, and detects what is arguably one of the most commonly performed tampering schemes: image splicing that breaks the original grid alignment either due to its placement or due to resampling transformations (scaling, rotation, etc.) of the spliced area.

The proposed method extends a JPEG grid detection algorithm from the literature [3] by introducing two novelties:

[1]Splicing occurs when parts of the original image are replaced by alien content. Together with inpainting and copy-moving, they constitute the most common types of forgery.

- a grid alignment confidence measure designed to identify whether an image block violates the global grid pattern, either due to misalignment, distortion, or complete absence of encoding artifacts (Section 3);

- a content-aware filtering step designed to account for grid discontinuities caused by the image content, strengthening the method's localization ability and overall output interpretability (Section 3.3).

The proposed method, hereafter referred to as CAGI (Content-Aware detection of Grid Inconsistencies), is evaluated against several state-of-the-art algorithms on three publicly available datasets, including both synthetic and real-world tampering cases. We test its classification ability, its localization effectiveness, and the readability of the produced outputs. The experimental results highlight the method's robustness over the diverse tampering scenarios and its contribution in terms of successful localizations of unique cases, i.e. cases that all other methods failed to detect. Java and MATLAB implementations of CAGI have been made publicly available as part of our Image Forensics Toolbox[2], alongside other state-of-the-art algorithms.

2. Related Work

Many notable contributions have been made towards tackling diverse cases of image manipulation. One category of approaches includes algorithms based on machine learning, using appropriate features extracted from images and trained on samples of tampered and authentic images [4, 5, 6, 7]. Others detect operation-specific traces (such as re-sampling) [8, 9], make use of compression and coding artifacts [10, 11, 12], search for inconsistencies in the image traces produced by the capturing process [13, 14], and search for physical inconsistencies such as illumination discontinuities [15, 16]. A number of surveys present the evolution of the state-of-the-art through time [17, 18, 1, 19, 2]. Here, we focus on methods for image splicing, organized by the type of trace they attempt to analyze for detecting forgeries. For each method, a three- or four-letter abbreviation is also given and used throughout the paper, following the conventions of [2].

Methods based on JPEG attributes: The method in [10] (BLK) is probably the most closely related to ours, since it also attempts to detect forgeries by

[2]https://github.com/MKLab-ITI/image-forensics

locating inconsistencies in the JPEG blocking artifact. The image is filtered based on local derivatives, weak edges are detected, and their conformance with an aligned 8×8 grid is measured. A feature corresponding to the local strength of the blocking pattern is extracted. The feature's variations indicate local absence or misalignment of the grid, which can be considered an indication of tampering. In [11] (ADQ1), tampering localization is achieved by exploiting the characteristics of double Discrete Cosine Transform (DCT) quantization. When splicing an object on a JPEG image, the spliced region often loses its JPEG traces, due to rescaling, rotation, filtering, or other transformations. Thus, when resaving the forged image, the unspliced part will exhibit the traces of two compressions, while the spliced part will only have undergone one. Recently, in [20] a novel approach is proposed, where convolutional neural networks are used to compute DCT coefficients and their histograms, and used to separate single from double compression. Experiments are run on pixel values, noise residuals, and DCT coefficients estimated from the image, and a window-based approach is shown to be promising with respect to tampering localization.

Methods based on DCT coefficients: In [21] (DCT), a fast detection method looks for inconsistencies in JPEG DCT coefficient histograms. The method in [22] (ADQ2) first estimates the quantization table used by the first JPEG compression and then attempts to model DCT coefficient histogram periodicities. The method in [23] (ADQ3) performs Aligned Double Quantization inconsistency detection using SVMs trained on the distribution of DCT coefficients for various cases of single vs double quantization. The method in [23] (NADQ) searches for Non-Aligned Double Quantization traces, that is, cases where the JPEG grid has been shifted prior to the second compression. Finally, in [24] (GHO) the image is recompressed at multiple different quantizations and subtracted from the original, aiming to detect JPEG Ghosts, i.e. traces left in the image for which past recompressions were performed at different quality compared to the unspliced image.

Methods based on CFA interpolation pattern disturbances and noise patterns: The method in [14] (CFA1) looks for disturbances in the image Color Filter Array (CFA) interpolation patterns left by the image capturing process by modelling them as mixtures of Gaussian distributions. The work in [13] presents two algorithms (CFA2 and CFA3) also exploiting CFA patterns: the first emulates the CFA filtering process and localizes regions that diverge from the expected result, while the second isolates image noise using de-noising, and compares noise variance between interpolated and natural

pixels. Finally, notable approaches based on noise information include the method presented in [25] (NOI1), where the local image noise is isolated by wavelet filtering and local variance discrepancies are treated as indicative of tampering, [26] (NOI2) where the local image noise variance is modeled using the properties of the kurtosis of frequency sub-band coefficients in natural images, and [27] (NOI3), where, following extraction of the high-frequency residual using a high-pass filter, the information is modeled using a co-occurrence descriptor, and inconsistencies in the local statistical properties of the descriptor are used to detect spliced regions. A more recent approach [28] uses PCA-based noise level estimation, coupled with k-means clustering and and adaptive block segmentation to identify splices. Another relevant work is [29], where, besides analyzing the local noise variance, the local texture inhomogeneity is also estimated, since it tends to misguide the noise algorithm. The authors show that by taking the local inhomogeneity into account, tampering localization performance can be increased. In [30] a different approach is followed, where an autoencoder is trained over steganalytic residual noise features, and local patches that do not conform to the learned model are labeled as tampered. Finally, in [31] a deep network is trained to extract noise residue information from an image and apply patch-based classification to localize tampered regions in an image.

Compared to the state-of-the-art, the proposed method (CAGI) aims to provide a tampering localization solution designed for robustness in realistic scenarios, while producing output maps that are easy to interpret. We specifically aim to achieve tampering localization for cases where the history in terms of acquisition, forgery, and post-forgery transformations of the images is unknown. The algorithm does not require metadata, JPEG compression parameters, or prior knowledge on the history of the image, nor does it require that the image is in raw format taken directly from the camera. It can operate on any file format, provided it has been compressed as JPEG in its past. The discrimination of the image areas that are aligned to the dominant grid pattern from those that break it is conducted through exhaustive search, taking also into account the contents of the image and their possible interference with the attempted modeling. This allows filtering out false activations and leads to overall cleaner outputs.

As will become apparent from the experimental study of Section 5, CAGI offers a higher level of versatility and overall performance compared to the state-of-the-art.

3. Method Description

Blocking artifacts appear as a regular pattern of visible block boundaries in a JPEG compressed image as a result of DCT coefficient quantization and the independent processing of the non-overlapping 8×8 blocks during the DCT. They are prominent in highly compressed images or images that have undergone multiple re-compressions, and become more subtle as the compression quality factor (QF) increases. These artifacts ultimately lead to the formation of a *block grid* in the JPEG image bitmap, i.e. a pattern of weak horizontal and vertical edges recurring every 8 pixels, starting from the upper left corner of the image.

As a first step, our approach improves upon the grid position estimation approach presented in [3], by adding a secondary level of analysis which allows us to estimate the grid position more reliably. [3] proposes estimating a measure K for each candidate grid position, and picking the position with the highest K. We propose a measure K'' drawn from the interrelationships between values of K at different positions, which is much more robust with respect to grid anomalies. To estimate K'', we first estimate two intermediate measures: K', which calculates the value differences between values of K at different positions, and S which analyzes the sign patterns of K. K'' is then calculated as a combination of K' and S.

Consecutively, we mark the blocks that do not conform to the detected grid indicated by K'' as tampering candidates. However, the absence of grid conformance in a region may not necessarily be the result of tampering. Instead, the visual content of the image may interfere with the grid detection process. Such cases include image areas that i) contain strong edges (artifacts appear around high-contrast edges producing a "halo" effect), ii) overexposed areas (where the soft grid pattern completely disappears), iii) underexposed areas (where the pattern is noticeably more subtle), and iv) textured areas containing patterns that resemble a grid. Furthermore, normal sensor noise introduced during image acquisition or any type of noise embedded in the image may also hinder the grid detection. Thus, we exploit the maps calculated during the first step, combining them with other post-processing operations, to generate a series of intermediate maps which are then fused into the final algorithm output.

The following sections provide a detailed description of the various steps involved in the proposed method.

3.1. Estimation of JPEG grid position

To detect the JPEG block grid, we extend the method proposed by Fan et al. [3]. The original intention of their work was to determine whether an image had been previously JPEG compressed and estimate the previous compression parameters. To do this, the method attempts to detect whether a JPEG grid pattern appears in the image, aligned at position (4,4) and repeating every 8 steps. The method evaluates inter-pixel differences over certain crucial positions in the block -essentially, the differences of pixel values *within a block* and *across block boundaries*. In [3], the image is split into N non-overlapping 8×8 pixel blocks and for each block(i, j) the scores $Z'(i, j)$ and $Z''(i, j)$ are computed using Equation 1.

$$Z'(i,j) = |A - B - C + D| \text{ and } Z''(i,j) = |E - F - G + H| \quad (1)$$

where A-H refer to pixel positions on a block as depicted in Figure 1. Then, two normalized histograms H_I and H_{II} are created from the Z' and Z'' scores across the image, and a confidence score K is computed using Equation 2.

$$K = \sum_{m=1}^{M} |H_I(m) - H_{II}(m)| \quad (2)$$

where M is the number of histogram bins used in the implementation (see [3] for further details). Fan et al. [3] empirically found that for pixel values ranging from 0 to 1, $K > 0.25$ is an indicator of successful grid detection. We will be referring to the detected grid position using the coordinates of pixel A in block$(1, 1)$ (Figure 1). According to this convention the default Grid Position (GP) for an unchanged JPEG compressed image should be located at $GP(4, 4)$. In case the grid has been shifted from its original position, e.g. due to image cropping, an investigation can be conducted by calculating and finding the highest confidence score K for all possible coordinates of pixel A within the 8×8 block (the coordinates of pixels B-H change accordingly, keeping their relative positions).

Figure 2 provides more insight into the matter by illustrating four distinct instances (a-d) of the grid localization process. More specifically, with the correct grid position being at $GP(4, 4)$, instance (a) is expected to produce the highest K score. Indeed, in case (a), as can be seen in the respective histogram plot, the majority of inner-block sampled pixels (A-D, H_I) have low Z' scores, while border pixels (E-H, H_{II}), score higher in terms of Z'', which maximizes Equation 2.

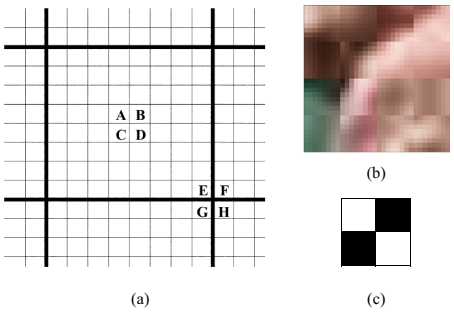


Figure 1: a) Depiction of JPEG 8×8 grid (bold lines) over image pixels. Pixels labeled A-D are sampled periodically to represent the inner part of the grid, while E-H are sampled on grid intersections [3], b) 32x32 JPEG image displaying visible grid artifacts, c) Pixel intensity pattern (cross pattern) maximizing Z' and Z" (Equation 1).

In instance (b), all neighbouring pixels are actually sampled from inner-block regions, completely failing to detect the grid position, clearly reflected also in the histogram plot. Even though not included in this example, the same goes for sampling only from border regions (e.g., $GP(8, 4)$). Instance (c) depicts a detection attempted at position $GP(5, 4)$. The inner-block and border pixels are somewhat correctly sampled i.e, pixels A-D are still within the inner-block region of the grid pattern while the border samples miss the grid intersection point by only one pixel in the vertical direction and thus partially meet the cross pattern (Figure 1.c). As a result, the respective histogram plot is very similar to the one of case (a), but the respective K detection score will be lower. Finally, instance (d) showcases the symmetrical properties of the applied computations. A position search for A(8,8), produces an identical plot as in case (a), only here, H_I and H_{II} are inverted, as is the sign of $H_I - H_{II}$.

According to [3], the highest K should reveal the grid position.

However, in our experiments with tampered and untampered images, K was found to be a poor grid location indicator, mainly because periodically sampling to detect the pattern could be heavily affected by image content, especially for images of low resolution (small total number of blocks) or high quality compression (weaker grid pattern) and even more so for tampered images where entire regions are misaligned with the main grid due to splicing.

In order to limit the possibility of high K scores being a result of image content, we propose adding a secondary level of analysis, examining the in-

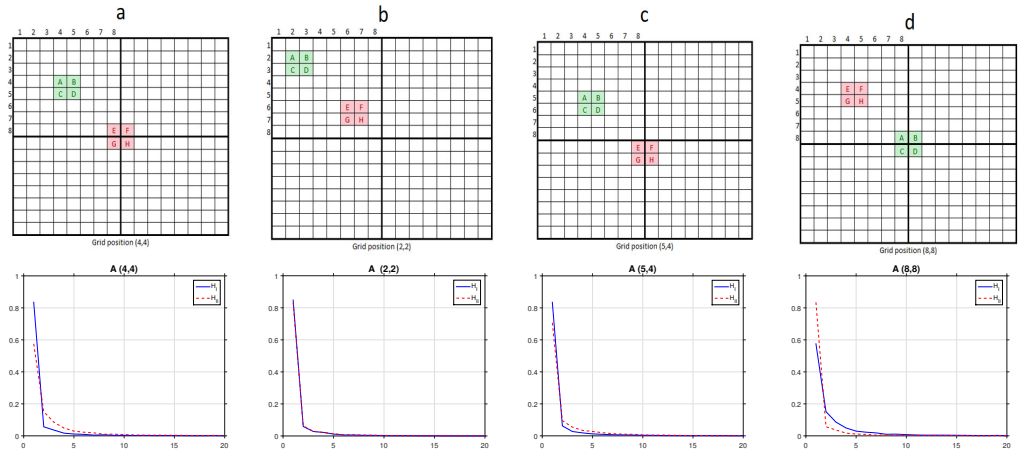


Figure 2: Illustrative examples of four different instances (a-d) evaluated during the grid localization process.

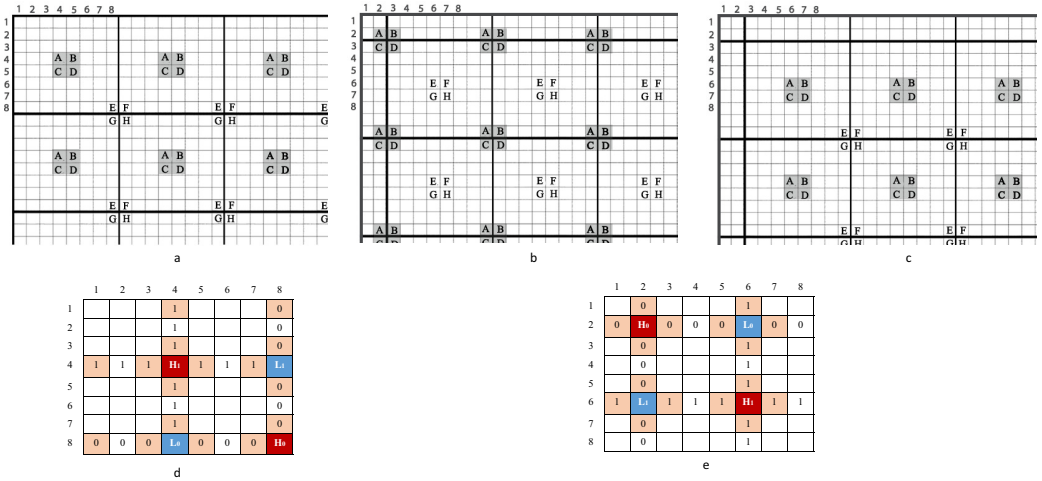


Figure 3: Visualized bitmap examples of grids at position a) $GP(4,4)$, and b,c) $GP(6,6)$, with marked sampling instances that present the highest K scores. Expected K-score patterns for d) $GP(4,4)$, and e) $GP(6,6)$.

terrelations between values of K at different candidate grid positions. This leads to a new grid confidence measure, namely K''. This new confidence measure does not simply rely on the highest reported K score to locate the grid position but includes an additional verification step based on the expected pattern, arising among all calculated K scores, that should be present at the correct grid location. Thus, our approach looks for a specific pattern in the values of K instead of simply taking the location where it is highest. Furthermore, it goes beyond the values of K to also analyze the symmetry of the histogram patterns. In its original formulation shown in Equation 2, it does not matter which histogram has more values at high bins and which one has more at the low bins, but only their absolute difference. However, knowing which term out of Z' and Z'' has higher values (i.e. samples located at the boundary) and which one has low values (i.e. samples at non-boundary positions) is also important for localizing the grid. Thus, besides calculating the K value at each candidate grid position, we also retain a "sign" for the position, with value 1 if Z' has more low values than Z'' (i.e. if A is located in an internal block position and E is alongside the boundary), and value 0 if the opposite is true.

We then exploit the spatial patterns of K scores and this "sign", to locate the grid more robustly without being distracted by potential isolated local maxima of K. More specifically, the expected pattern suggests that if the highest K score is found at position (i, j), then an equally high K score should be present at position $(i + 4, j + 4)$, and low scores at positions $(i + 4, j)$ and $(i, j + 4)$. Furthermore, the K scores of different GP investigations remain high and positive as long as A-D are actually part of the inner block, while E-H are at the borders, or high but with a zero sign, if sampled inversely. If pixels are sampled being all in the same class (all inner-block or all border pixels), the respective K scores are expected to be low and their sign uncertain. Figure 3 demonstrates this emerging pattern. Figure 3.a illustrates a grid at position $GP(4, 4)$ and the sampling instance (out of all possible 64) that will produce the highest K score with the correct sign. Figure 3.d shows the respective K-score patterns during the grid location investigation. Letters H and L stand for High and Low K scores, respectively. Positions marked with 1 indicate that the inner pixels A-D are correctly part of the inner region of the grid and E-H are along grid boundaries. Positions marked with 0 indicate the opposite. Thus, after we calculate the values of K and locate the sampling instance that produces the highest one for a given image, we include an additional step in which we also evaluate if the rest of

the calculated K scores and their signs comply with the expected pattern. In Figures 3.b and 3.c the grid is shifted by two pixels in both directions. The grid detection process will locate the grid at $GP(6,6)$ (Figure 3.c) and the expected K score pattern will be adjusted as shown in Figure 3.e. Due to the symmetry of the sampling process, however, the investigation instance at position (2,2) (Figure 3.b) will also produce the same absolute K score (with an opposite sign), as well as the same K scores pattern (Figure 3.e).

The final grid estimate is based on a combination of K value patterns, expressed by an intermediate confidence score K', and sign patterns expressed by a measure S. K' is calculated based on Equation 3.

$$K'(i,j) = \frac{K(i,j) + K(i+4,j+4) - K(i+4,j) - K(i,j+4)}{4} \tag{3}$$

where $K' \in [0,1]$. The value of $K \in [0,2]$ is calculated by Equation 2. The aim of K' is to quantify the observed patterns in the values of K. Leveraging the pattern symmetry of K (without the signs), we may reduce the investigation of possible grid positions from 64 (8-by-8 window of positions) to just 16 (4-by-4 window) and identify the actual position by comparing the sign of $K'(i,j)$ to those of the original $K(i,j)$ and $K(i+4,j+4)$.

For the sign patterns, we also evaluate these 16 grid positions with respect to how well they match the expected pattern (see Figure 3, where 1 and 0 indicate positive and negative signs, respectively). Starting at position $GP(i,j)$ and searching horizontally and vertically, most K signs should be positive, while for position $A(i+4,j+4)$ most should be negative.

A measure $S \in [0,1]$ is used to evaluate how well each position fits this pattern, calculated as the number of positions having the expected sign given the candidate grid, divided by the total number of positions.

Then, the final confidence score is formulated as the mean of the K pattern estimate and the sign pattern estimate, as indicated by Equation 4.

$$K'' = \frac{1}{2}(K' + S) \in [0,1] \tag{4}$$

K'' is a score referring to the total image and its aim is to estimate the position of the JPEG grid. Once the position of the grid is fixed, we can also calculate the contribution K''_{block} of each individual 8×8 block to the overall K'' score. The calculations for K''_{block} follow that of K'', only instead of using the normalized histograms of all sampled pixels of all blocks to calculate K (i.e. Equation 2), we compute individual K_{block} scores for each block n as:

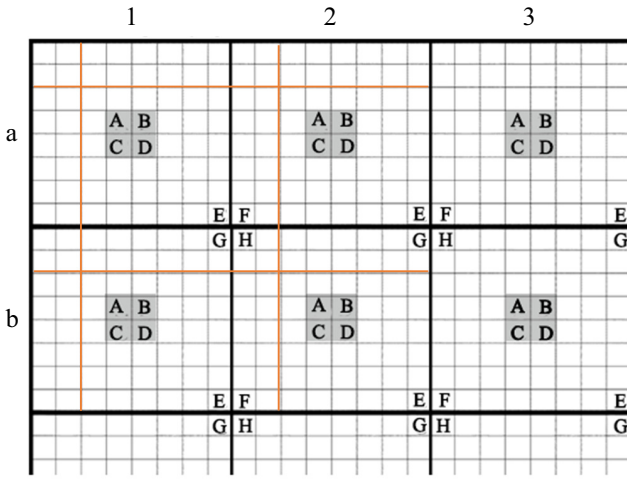


Figure 4: Example of multiple JPEG grids. Black grid is the original grid and orange grid is introduced by a tampering operation, e.g. splicing.

$$K_{block}(n) = Z'(n) - Z''(n) \tag{5}$$

and proceed with the calculations as above, to get the respective $K''_{block}(n)$.

K'' takes advantage of the lightweight implementation and effectiveness of the K measure and adds an extra level of detection robustness, while $K''_{block}(n)$ allows the identification of image parts that present JPEG grid inconsistencies, which is the goal in detecting and localizing tampering operations. In Figure 4, for instance, the image blocks $a1, a2, b1$ and $b2$ present traces of two different grids (black for the original and orange for the result of misaligned image splicing), while blocks $a3$ and $b3$ carry only the original JPEG artifacts. The individual K_{block} scores would not reveal the inconsistency because the sampled pixels do not happen to belong to both grids. K''_{block} however, would result in lower scores for the four tampered blocks compared to the untampered ones, since the expected pattern will not be equally strong in the respective K_{block}-score pattern and sign evaluations.

3.2. Localizing grid discontinuities

The steps of our approach so far have allowed us to detect the presence of a JPEG grid, and estimate its alignment -which, granted, will in most cases be located at position (8,8), but cropping the image may result in it being shifted. It has provided us also with local estimates of the contribution of

each block to the final estimate, as calculated by Equation 5, which can be used to localize the forgery.

Consider the following typical case of splicing, in which a host image is JPEG compressed, an alien region is cut from another JPEG image, pasted into the host (not aligned with the original grid) and the composite image is re-compressed as JPEG. At the location where the tampering took place, the new image bitmap will carry overlapping grid artifacts.

In the ideal case, where the grids' positions of the original host image and the one caused by the final compression are known and can be detected using K'', we would only need to plot the heat map of the contribution of each 8×8 block to the maximization of K'' for $i = 4$, $j = 4$ (standard grid position of JPEG). Blocks ranging significantly low would correspond to local inconsistencies in the main grid pattern, caused by the alien region. Unfortunately this is hardly ever the case, since the consistency of the blocking artifacts throughout the host image is easily disturbed from a variety of factors, such as strong edges, visual texture patterns, over/under exposure, etc.

To moderate the impact of such effects, we exploit all information gathered during the previous procedure. Besides the heat map of local block contributions to K'', we also produce a series of auxiliary maps aimed to isolate and remove the artifacts produced by such phenomena, and only keep the regions that we can confidently assume that are violating the JPEG grid due to tampering.

To produce these masks we exploit: a) the heat map of the local K''_{block} scores for the best-fitting grid; b) the heat maps of the local K''_{block} scores for all other candidate grids; c) an edge detection map used both to remove strong edges from the results (as they tend to disrupt false positives) and to locate soft, widespread edges, (which we use as indicators that the block is suitable for accurate grid estimation); and d) a map identifying over- and under-exposed areas, where grid detection would be impossible anyway, and thus any inconsistencies found there are unreliable.

Figure 5 presents an overview of the method. A series of Heat and Help Maps are built and combined to produce the final output. In Figure 5 we use an image example as input, and visualize the intermediate stages up to the final output. The input image is taken from the Fontani et al. Synthetic dataset (Class 4) [32] with the tampered area marked by the semi-transparent, yellow-outlined rectangle in the initial image.

With respect to information types a) and b), for each image block we calculate the rectified K''_{block} scores for the 16 possible grid coordinates (Equation

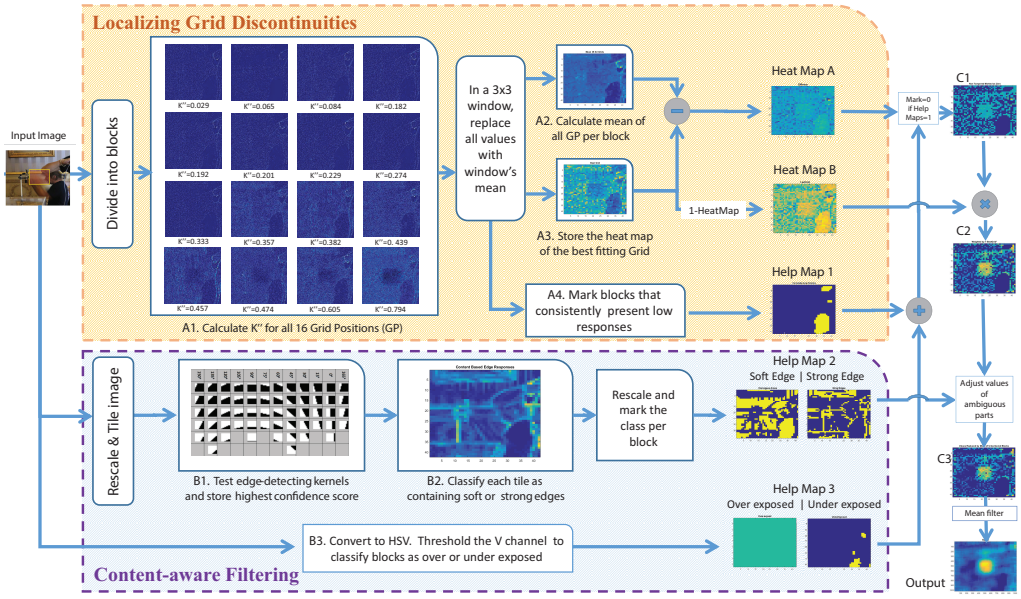


Figure 5: Overview of the proposed method with visualized example results of the intermediate stages and final output.

6), and then compute the mean block response (Equation 7).

$$\text{fit}(x) = H[K''_{block(i_x,j_x)}] \cdot K''_{block(i_x,j_x)}, x \in [1, 16] \tag{6}$$

$$\text{fit}_{BLK}(n) = \frac{1}{16} \times \sum_{x=1}^{16} \text{fit}(x) \tag{7}$$

where $H[k]$ is the Heaviside step function, and (i_x, j_x) is the pair of coordinates for one of the 16 candidate grid positions within the block. We thus generate two maps, one containing the mean block responses for all 16 possible grid alignments, and one for the best-fitting grid alignment.

In Figure 5, the outputs (in the form of heat maps[3]) for six out of the 16 investigated grids for Equation 6 are depicted in Figures 5.A1. The upper row of A1 shows the outputs reporting low K'' while the lower row shows the higher scoring K'' grid position searches.

[3]All heat maps in the paper are based on MATLAB's `parula` colormap.

It can be seen in this example that, as we move from the least to the best fitting grid, the tampered region becomes visible as an area of low response values. In parallel, however, all grids, even the ones with low K'' scores, present strong responses at certain locations where the image content disrupts the result of Equation 5, mostly due to the presence of high-contrast edges. In a similar manner, weak responses can be found for all grids at under-exposed (dark) image areas and at bright image regions (upper right corner), where the grid pattern is more subtly present. The various Heat Maps and Help Maps we have devised are aimed to remove those effects and only keep the actual tampering trace. Figures 5.A2 and 5.A3 show the calculated mean responses of all 16 grids per block and the responses of the best detected grid, respectively. Both maps have been mean filtered with a small window size to remove spurious outputs.

One effective way to separate regions where the grid is actually broken from those where the grid is made undetectable due to content, is to look at the mean response: if a region has low response for all alignments, it is most likely due to content and not due to misalignment to a specific grid. We want to suppress these responses, thus we take the difference between the mean response and the response of the best detected grid. This gives us Heat Map A (Figure 5), where many areas with undetectable grids are suppressed while areas of grid pattern discontinuity are emphasized. The subtraction of A2 from A3 has the additional effect of resulting in Heat Map A having high values in candidate tampered areas and low values in untampered areas.

A second intermediate map is Heat Map B (depicted in Figure 5), aimed to be used later as a weighting factor in characterizing blocks as tampered or not. It is produced by inverting the best fitting grid map, so that locations of grid inconsistencies return high responses. In this sense, Heat Map B contains the base result of the grid inconsistency detection algorithm.

3.3. Content-aware filtering

While Heat Map A was produced by removing misguiding regions after identifying those blocks that did not contain a detectable grid in any alignment, the output is far from easy to interpret. It is evident by examining the heat map (Figure 5, Heat Map A) that an inexperienced user would have difficulty assessing the location of the actual tampering by inspecting the map. In an effort to produce more reliable and interpretable outputs, we proceed with an extra computational step of coarse image segmentation based on image content. There are four types of image content we wish to be able to detect in order to analyze and filter the initial output:

1. homogeneous areas, i.e. areas where the intensity gradient between neighboring pixels is near-zero,
2. over/under-exposed areas,
3. areas of high-edge contrast, and
4. areas of soft edges.

With respect to homogeneous areas (point 1), the problem is that, when JPEG encoding is applied on image parts of near-uniform colors that span multiple image blocks, the grid pattern is exceptionally weak or non-existent even for low quality encodings. Thus, we need to determine whether grid discontinuities (including complete absence or significantly weaker artifacts) are signs of tampering or simply due to homogeneous areas. To this end we produce a specialized map, depicted in Figure 5 as Help Map 1, in which we mark blocks that score consistently low (near-zero) over all 16 GP.

Over- and under-exposed blocks (point 2) also make grid detection very difficult, and thus might mislead the detection algorithm. These blocks can be detected by converting the image into the HSV space and using upper and lower thresholds, respectively, in the Value (V) channel. In our implementation, we empirically found that mean values that are higher than 95% of the channel maximum possible value can be securely classified as over-exposed, while values lower that 5% can be classified as under-exposed. The result is stored in Help Map 3.

With respect to detecting areas of high-edge contrast and soft edges (points 3 and 4), the aim is to isolate regions containing "soft" edges, i.e. edges that are strong enough to create content variance and allow grid localization, but not so strong as to disrupt the localization algorithm. Regions characterized by soft edges can be considered the most representative in terms of the grid fitness scores they produce. We need a representative value to use for the regions that we marked as untampered/unsuitable for detection using the Help Maps. This value needs to be low enough to allow tampered regions to stand out, but not so low as to end up highlighting the rest of the image. Localizing regions of soft edges and getting their mean fitness provides a dynamic way to get such a value, which will allow us to produce output maps that are not only accurate in terms of localization, but contain enough contrast between tampered and untampered regions to be easily readable by an untrained human investigator.

We employ a novel efficient edge extraction scheme inspired by [33] that is able to adaptively classify the detected edges as salient or soft. To en-

sure consistent computational times and results, the input image is resized to the largest dimension scaled to 960 pixels (the smallest is scaled near-proportionately, but ensuring it is a multiple of 8, to allow block-based tiling and filtering).

This step only serves to identify the softly textured portions of the image, so as to use their average K" values as a reliable baseline. Rescaling will not change this property of any image region. Since this step is not essentially linked to any forensic operation, we do not have to worry about destroying sensitive traces. Rescaling will not result in any loss of relevant information, but will save us significant computation time.

The rescaled image is then tiled into non-overlapping 8×8 blocks that are independently processed by a set of 2-dimensional 8×8 edge detection kernels. The kernels are an adaptation of the kernel masks presented in [33]. In our implementation, the kernels are binary masks consisting of two regions (a dark and a light), defining edges in 12 orientations on 15° increments. For each of these orientations, an appropriate number of instances represents all possible positions (2-pixel shifts) of the edge within the region of the kernel, resulting in a total of 58 kernels (Figure 5.B1).

Each image block $B(i, j)$ is then processed by all 58 kernels in order to calculate an edge confidence score based on Equation 8.

$$C_z = \left| \sum_{i=1}^{8} \sum_{j=1}^{8} B(i,j) \cdot \bar{k_z}(i,j) \left[\frac{1}{M_w} - \frac{1}{M_b} \right] \right| \in [0,1] \tag{8}$$

where M_w and M_b the number of white and black pixels in the kernel, respectively, and $\bar{k_z}(i, j)$ is the bitwise NOR for position (i, j) of kernel $k_z, z \in [1, 58]$.

When all blocks have been processed by all kernels the highest confidence score is stored for each block. To discriminate block edge responses into salient or soft, a thresholding step takes place. The image is divided into six areas (A-F), each of which is further divided into six sub-regions $(a_1, a_2...a_6, b_1, b_2...b_6, ..., f_1, f_2...f_6)$ as illustrated in Figure 6. To determine a threshold value for each one of the smaller regions (second level regions), we calculate (i) threshold T_{img} to be the mean confidence score over the whole image, (ii) thresholds $(T_A, T_B, .., T_F)$ to be the mean confidence scores of the tiles belonging to each first level region, and (iii) $(T_{a_1}, ...T_{a_6}, T_{b_1}, ...T_{b_6}, ..., T_{f_1}, ..., T_{f_6})$ to be the mean confidence scores of each second level region. Then, the threshold for each second level region is selected to be the largest among

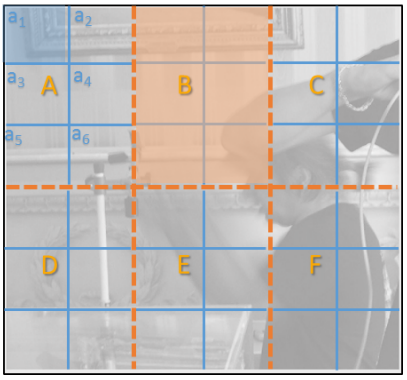


Figure 6: Image partitioning used for the determination of local edge thresholds.

the one calculated from the second-level region, the one calculated from the containing first-level region, and the overall image threshold. For instance, in the case of sub-region a_1, we would set $T'_{a_1} = \max(T_{a_1}, T_A, T_{img})$.

This thresholding process is important because it evaluates strong edges, not by an absolute number but locally, taking into account local image statistics. Applying the thresholding is crucial for the quality of the output maps, because these maps have scaled value ranges: this means that, in the absence of high-contrast edges, low-strength edges would be dominating the output heat map and would falsely indicate possible forgery. The proposed adaptive thresholding scheme scales the produced thresholds in relation to the overall contrast of the content and overcomes the issue.

The bottom part of Figure 5 illustrates the content-aware filtering part of the method. Specifically, Figure 5.B2 depicts the color-scaled illustration of the highest confidence scores C_k per block. Help Map 2, depicts the example maps resulting after the classification of the blocks as containing soft and strong edges, respectively. The first map presented under Help Map 3 shows the map of under-exposed blocks and the second, being flat, informs us that in this particular image no over-exposed blocks were found.

3.4. Creating the final output map

The final step of the method aims at producing a readable output, with clear contrast between tampered and untampered regions. To this end, it utilizes all intermediate information, i.e Heat Maps A,B and Help Maps 1-3 (Figure 5). Heat Map A contains the grid discontinuity detection results with the non-relevant regions suppressed, while Heat Map B contains the output

of the best matching grid discontinuity detection, and is intended to be used as a weighting factor that will highlight the non-conforming regions.

In Heat Map A, blocks with high values generally result from over/under-exposed image regions, homogeneous regions or tampered regions, while blocks with low values are most likely unsuppressed responses of strong edges. Since the tampered region is expected to exhibit high values, we mark all blocks with values lower than the heat map mean as non-tampered. Next, we use Help Maps 1 and 3 to also mark homogeneous and over/under-exposed blocks as non-tampered. The visualized output of this process is illustrated in Figure 5.C1.

The resulting map is then weighted by Heat Map B (i.e. the inverse heat map of the best grid) resulting in the heat map depicted in Figure 5.C2. This map could itself serve as the final output of the algorithm, as the highest values are expected to correspond to the tampered region. However, the issue remains on what value to assign to the blocks marked as untampered, so as to create a human-readable map with an easily visualizable value range where the tampered regions will stand out. Assigning zeros is not an ideal option because heat map visualizations are always relative in scale. Thus if the original map values were high, the presence of zeros may result in an output that is almost binary, with zeroed regions on the one end, and all other blocks, tampered and untampered alike, on the other. To mitigate this issue, at the final step we replace all marked blocks with the mean value of those soft edge blocks (Help Map 2) that are not classified as homogeneous (Help Map 1). We experimentally found this value to serve as a good approximation to the value range of untampered, non-zeroed regions. Zeroed and non-zeroed untampered block values are now brought to roughly the same range (Figure 5.C3), which should make the tampered region visually stand out in the heat map. The final output map is produced by mean filtering (Figure 5.C3).

Figure 7 showcases the importance of the two introduced novelties, i) the stronger confidence measure K'' employed to identify whether a block follows the global grid pattern or violates it, and ii) the content-aware filtering stage employed to suppress false activations deriving from image content. By comparing the outputs produced by the proposed method (fourth row) with those produced when leaving out either of the two proposed novelties, i.e. the newly proposed grid alignment confidence measure (second row) and the content-aware filtering step (third row), it becomes clear that the accuracy and quality of the output maps improves considerably.

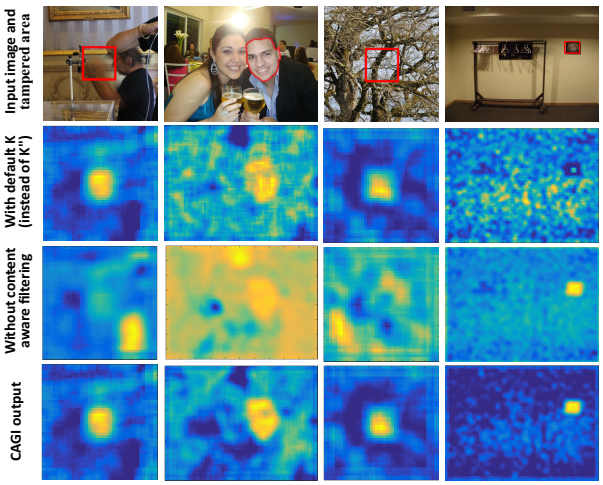


Figure 7: Examples showcasing the contributions of CAGI. *First row:* input images where the tampered area is marked with a red outline. *Second row:* Outputs maps produced without the newly proposed grid alignment confidence measure. *Third row:* Output maps produced without the content-aware filtering step. *Fourth row:* Output maps produced by the CAGI method.

3.5. Inverse discontinuity detection

The proposed method, as described in Section 3, assumes that the discontinuities will appear as areas of lower responses, in terms of K'', in relevance to the rest of the image's responses, during the search for the best fitting grid. The relative strength of the responses is, however, very much affected by the compression Quality Factor of the host (QF_h), the QF of the alien splice (QF_s) and the final compression QF of the composite image (QF_f).

Consider, for instance, the following scenarios: i) QF_h is high (weak artifacts), the splicing comes from an image with $QF_s < QF_h$ and for the final compression quality, we have $QF_f > QF_h > QF_s$, and ii) the host image is compressed losslessly (QF=100), the splice is JPEG compressed, and the composite image is again saved in lossless format.

In both of these cases, discontinuities will appear as areas of high K'' response in relevance to the overall low responses calculated in the image. That is, the algorithm will locate a grid only on the spliced area, and assign low K'' values to the rest of the image. Due to the inversion step prior to forming Heat Maps A and B, and combined with the mean value substitution step of Figure 5.C3, in those cases the algorithm will most likely not be able to localize the splice. Some algorithms, particularly ones based on noise

or block artifact discontinuities handle this by shifting the burden of the interpretation to the human analyst. In such algorithms, the tampered area may appear either as a region of disproportionately higher or lower response. CAGI, however, has an integrated post-processing step that both aims to increase detection accuracy and to produce more human-readable outputs. Thus, it is necessary to adjust the algorithm to this eventuality and treat this sub-case in a targeted manner.

In order to account for cases like these, we introduced an additional branch to the method that produces a complementary output map. More specifically, at the last stage of the algorithm when extracting the final output map, instead of filtering (marking as zero) the blocks in Heat Map A that range under the map's mean, we now filter those that range over that value. As before, we also mark the homogeneous and over/under-exposed blocks and proceed by assigning the mean value of the soft edge blocks (Help Map 2) that are not classified as homogeneous (Help Map 1).

The complementary output produced by this straightforward, inverted filtering can be presented to end users along with the original output. This will allow them to choose the most appropriate result based on visual inspection. We refer to this output as *inv-CAGI*.

4. Evaluation

We evaluate CAGI through a number of experiments, which provide insight into its potential for i) blind tampering detection and ii) localization, also evaluating iii) the method's output interpretability and iv) robustness against common post-processing operations and in realistic datasets, where details concerning the image history and applied transformations are unknown. For all employed datasets and scenarios we additionally provide results concerning the number of achieved detections of high confidence and the method's contribution in terms of detecting unique cases.

With that said, the proposed method is directly compared to seven methods from the state-of-the-art (Table 1). With respect to the methods described in Section 2, ADQ1 was selected to represent approaches that base their detection on double quantization. ADQ1 has the advantage of being able to operate on images that had been compressed as JPEG, and were then decompressed and stored in PNG, which is the case in some datasets. In contrast, ADQ2, ADQ3 and NADQ can only operate using JPEG images as input, since they require specific information derived from the JPEG

file, such as the decompression rounding residue or the quantization matrix used for the last compression. Since some datasets contain PNG images which carry the traces of past JPEG compressions but have already been decompressed, that information is essentially lost and these algorithms cannot work. Also, GHO is not part of the selected methods because it produces several output maps per case, requiring thus manual investigation to trace the changes between the different maps to locate the forgery. Finally, given the expected limited applicability of methods that search for disturbances in CFA patterns, only results from CFA1 are presented as indicative of such methods.

Table 1: Overview of the selected state-of-the-art methods.

Acronym	Description
DCT [21]	Looks for inconsistencies in the JPEG DCT coefficient histograms to detect possible tampering.
BLK [10]	Identifies possible tampering by locating inconsistencies in the JPEG blocking artifacts.
ADQ1 [11]	Tampering localization is achieved by exploiting the characteristics of double DCT quantization.
NOI1 [25]	Models image noise using wavelet filtering and treats localized variances as possible forgeries.
NOI2 [26]	Models image noise using the properties of the kurtosis of frequency sub-band coefficients in natural images.
NOI3 [27]	Computes a local co-occurrence map of the quantized high-frequency component of the image and locates inconsistencies in the local statistical properties.
CFA1 [14]	Models the Color Filter Array interpolation patterns as a mixture of Gaussian distributions and locates tampering based on detected disturbances.

4.1. Datasets

Table 2 lists the employed datasets.

The first dataset employed in this study is the synthetic dataset by Fontani et al. [32]. This dataset will allow us to test the effectiveness of the method for controlled scenarios. It contains 4800 original and 4800 tampered images, which were generated by automatically extracting a fixed-size square from the center of the image and replacing it in the image, emulating the effects of a splice (e.g. removing the traces of JPEG compression, or changing the JPEG grid alignment). The tampered images of the dataset are split in four distinct classes, each containing a different type of forgery (Table 3). Thus, depending on the class, a forgery should theoretically be detectable

Table 2: Benchmark image datasets.

Dataset	**# Fake/Authentic**	**Format**
Fontani et al. synthetic [32]	4,800 / 4,800	JPEG
IFS-TC Forensics Challenge [34]	442 / 1,050	PNG (possible JPEG history)
Wild Web Dataset [35]	10,646 / 0	JPEG, PNG, GIF, BMP, TIF

by different combinations of Non-Aligned JPEG quantization, Aligned JPEG quantization and JPEG Ghost, while other algorithms may also be able to localize certain forgeries.

Next, we employ the First IFS-TC Image Forensics Challenge training set [34], a dataset containing user-submitted forgeries and their ground-truth masks. The dataset was designed to serve as a realistic benchmark (different types of tampering, unknown image history and possible post-tampering transformations). While images in this set are saved as PNG, it is likely most of them were originally in JPEG format, since they exhibit traces of past compressions (e.g. blocking artifacts or DCT coefficient histogram periodicities). Therefore, splices may be detectable using JPEG-based methods.

For the two aforementioned datasets, and despite the fact that the latter is considered to be a realistic benchmark, we also subject the images to rescaling (95%, 75%, 50%) and recompressing (90%, 70%) operations producing in total 5 variants of each original dataset. Since, tampering traces may disappear after common post-processing operations like resizing and resaving (which are operations applied automatically in many online image storing and sharing platforms, e.g. social media), these variants will allow us to conduct deeper experimental analysis of the method robustness.

Finally, we experiment with the Wild Web Dataset [35] that contains 78 cases of real-world forgeries. As the forgeries have been circulating various websites and social media platforms, there exist multiple versions of each forgery, due to resavings, croppings, and other transformations. The Wild Web Dataset was formed by collecting a large number of different versions from each forgery, resulting in a set of 10,646 images.

The uncontrolled, varying conditions under which the tampered images in this particular dataset were created, shared and collected will allow us to gain an additional level of insight concerning the robustness of the methods, which exceeds the limited tests and variations of post-processing transformations that we can manually subject the images to.

Table 3: Fontani et al. [32] synthetic dataset classes.

Class 1
Region is cut from a JPEG image and pasted, breaking the 8x8 grid, into an uncompressed image; the result is saved as JPEG.
Traces: Misaligned JPEG compression
Class 2
Region is taken from an uncompressed image and pasted into a JPEG image; the result is saved as JPEG.
Traces: Double quantization, JPEG ghost
Class 3
Region is cut from a JPEG image and pasted into an uncompressed image in a position multiple of the 8x8 grid; result is saved as JPEG.
Traces: JPEG ghost
Class 4
Region is cut from a JPEG image and pasted (without respecting the original 8x8 grid) into a JPEG image; the result is saved as JPEG.
Traces: Misaligned JPEG compression, Double quantization, JPEG ghost

4.2. Evaluation Metrics

Each of the tested methods produces an output map per image, in the form of a heat map, that can be used to detect forgeries and localize tampered areas. For tampered images, these output maps should have significantly distinct value assignments for pixels belonging to untampered image regions compared those belonging to tampered regions, while for authentic images the output maps should ideally be flat.

Tampering detection: For our first test, we evaluate the methods' ability to correctly classify tampered images based on the value distribution of the output maps following the methodology proposed in [35].

More specifically, the datasets provide binary ground truth masks for all tampered images, while an artificial ground truth mask is used for each untampered image similar to [24] and [32], which corresponds to a block of size 1/4 of each dimension, placed in the image center. The Kolmogorov-Smirnov (KS) statistic is used to compare the value distribution for the two regions of the masks (tampered/untampered).

$$KS = \max_{u} |C_1(u) - C_2(u)| \tag{9}$$

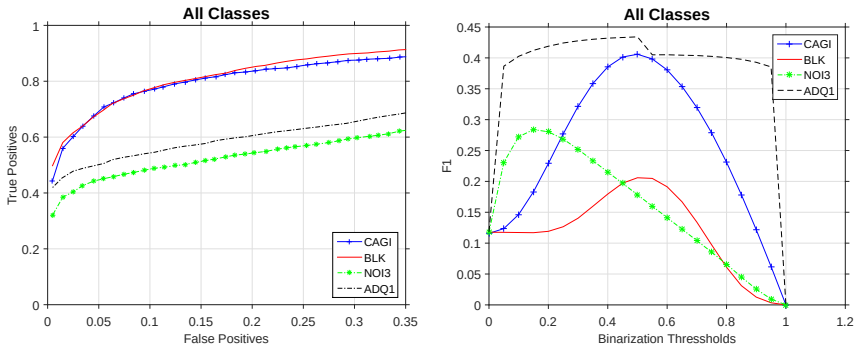
where $C_1(u)$ and $C_2(u)$ are the cumulative probability distributions inside and outside the mask, respectively. If KS surpasses a threshold, a positive detection is declared. ROC curves are calculated by shifting the threshold

for each algorithm, and evaluating how many images return positives in the tampered and untampered subsets. This methodology is appropriate for datasets that contain both tampered and untampered images, and sets a baseline against overestimation of a method's ability to localize tampering.

Tampering localization and output interpretability: Next we evaluate the methods, in terms of their localization quality and output readability based on the pixel-wise agreement between the reference mask and the produced output map of each method. For these tests only tampered images are evaluated, while the quality of the response is measured in terms of the achieved F-score (F1). This methodology requires the output maps to be thresholded prior to any evaluation. Since the range of values of the output maps for each algorithm varies, and in an effort to be fair, we first normalize all maps in the $[0, 1]$ range and proceed by successively shifting the binarization threshold by 0.05 increments, calculating the achieved F1 score for every step. The localization performance is presented in the form of F1 curves, while the readability of the output maps is related to the range of different binarization thresholds that yield high F1 scores, i.e. of at least 70% of the recorded maximum F1 score. F1 scores that remain high for a wide range of binarization thresholds indicate that the two classes (tampered regions/ untampered regions) have been correctly assigned distinctive enough values, such that interpreting the output would be easy for both human inspectors and unsupervised computer systems.

5. Experimental Results

This section includes the experimental results per dataset. To keep the presentation compact and to the point, we focus more on three of the reference methods that yield overall good results, while producing some of the most clear tampering localization heat maps. These include blocking artifact discontinuities (BLK), aligned double quantization (ADQ1) and SpliceBuster (NOI3). The experimental evaluation and comparison with the rest of the reference methods will be given more concisely in section 5.4, where we discuss the overall performance. Outputs in the form of heat maps produced by all methods employed in this paper, on various images from the realistic datasets, are available in Figure 16.



(a) (b)

Figure 8: Results in the Fontani et al. synthetic dataset over all classes: a) ROC curves, b) F1 score curves.

5.1. Results on the synthetic dataset by Fontani et al.

The dataset by Fontani et al. is synthetically generated, allowing to test the effectiveness of methods on different types of forgery. Figure 8(a) presents the experimental results using the first evaluation methodology over the whole collection. CAGI is overall one of the best performing methods together with BLK, achieving approximately 70% true positive rate at a 5% false positive rate.

It should be noted that NOI3, being a representative of noise-based algorithms, is not the most appropriate algorithm for this dataset. The Fontani et al. synthetic dataset was created as a means of evaluating JPEG-based algorithms, thus it deliberately includes forgeries which exhibit JPEG traces of tampering with minimal impact on content and noise.

Figure 9 presents the per-class results for the CAGI, BLK, NOI3 and ADQ1 methods. In Classes 1 and 4, where the tampered images carry traces of misaligned JPEG compressions, i.e the principle which CAGI is designed around, the method demonstrates competitive results and is only outperformed by ADQ1 in Class 4, where double quantization traces are also present. Interestingly, however, CAGI also manages to rank among the best performing methods for the other two classes.

Along with the class of tampering, a second factor comes into play concerning the robustness of the detection: the QF of the host image in relation to the final compression QF. The host images of this dataset were acquired in lossless format and (depending on the class) were compressed with varying compression qualities $QF_1(40-80)$. After the splicing operation, the

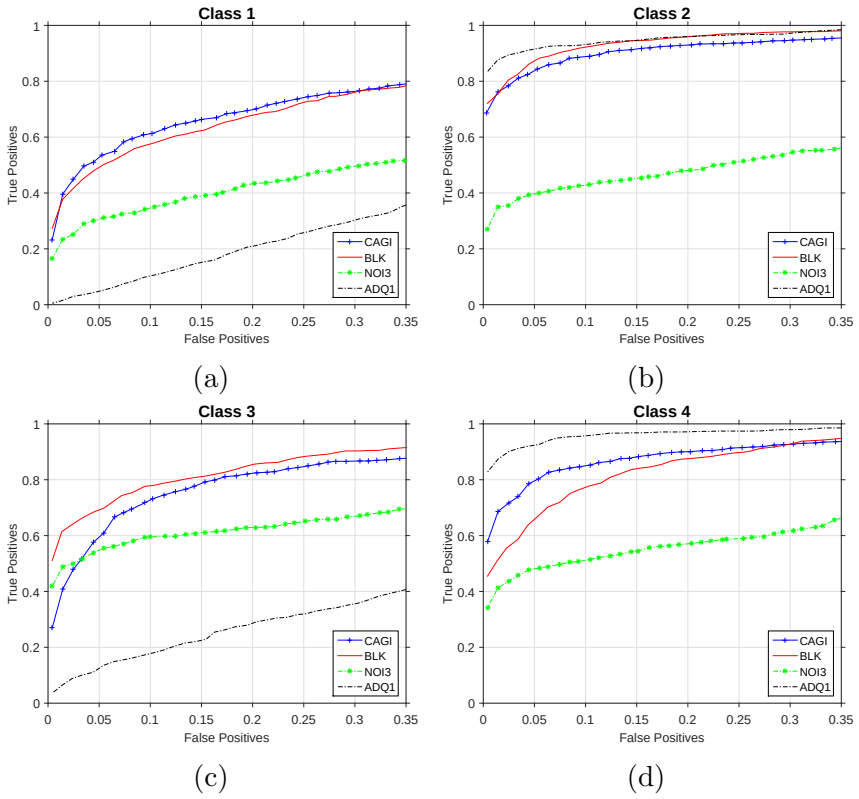


Figure 9: ROC curves in the Fontani et al. synthetic dataset per class: a) Class 1, b) Class 2, c) Class 3, d) Class 4.

resulting images are recompressed into JPEG.

In CAGI, discontinuities of the image grid appear as lower responding areas in the heat map of the best responding grid and the heat map of the mean response of all tested grids. Class 4, is completely in-line with CAGI's design. The misaligned JPEG splice can be generally traced easily. For Class 1, the localization of the misaligned patch is also relatively easy to achieve when the host is compressed with a low QF. However, as the QF of the final compression increases, the area that was initially uncompressed (host) only gets light artifacts after compression, while the double pattern within the spliced region is also degraded. This makes the detection vulnerable to responses derived from content.

CAGI is much more robust for tampered images of Class 2, since it can rely on artifacts that are already present in the host. The tampered area

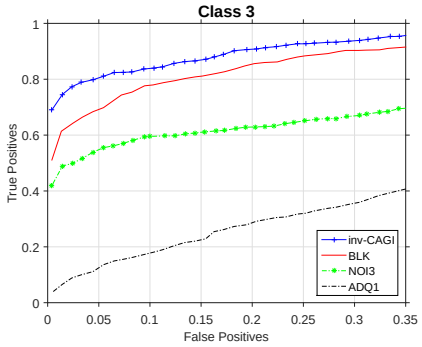


Figure 10: ROC curves in the Fontani et al. synthetic dataset Class 3: inv-CAGI, BLK, NOI3, ADQ1.

has lower responses due to the higher final QF_2 compression. Misses occur only in cases of extreme content-related responses that the employed content aware process fails to account for.

Class 3 is the most challenging for CAGI. The tampered patch is aligned to the grid created by the final compression and thus, for lower QF, both mean and best grid responses will highlight the tampered region with higher values. The method is this case is producing inversed maps, compared to what it was designed for. Depending on the QFs and the image content, this may not be an issue; tampered and untampered region will only appear inversed in the final output. In many cases, however, the operations that take place next, implemented with the intention of suppressing responses corresponding to high frequency content, may falsely treat the detection as edges. Inversed CAGI (inv-CAGI), as described in Section 3.5, was implemented to account for such cases, which are in fact quite common. The curves in Figure 10 attest the added value of the inv-CAGI variant of our method.

Moving on to the evaluations of the localization and readability quality of the maps, Figure 8(b) presents the mean F1 scores per binarizarion step over the whole Fontani et al. collection. The achieved localization is evaluated by the maximum mean F1 score for each method (at its respective best performing binarization threshold). CAGI achieves once more one of the best reported performances.

As discussed before, the interpretability can be evaluated based on the range of binarization thresholds where the achieved F1 remains high, as it suggests that the tampered and untampered image regions are characterized by significantly different values in the output maps. ADQ1, which produces

Table 4: Reported detections on Fontani et al. dataset for $F1_{score} >= 0.7$ and $F1_{score} >= 0.8$ at each method's best binarization threshold.

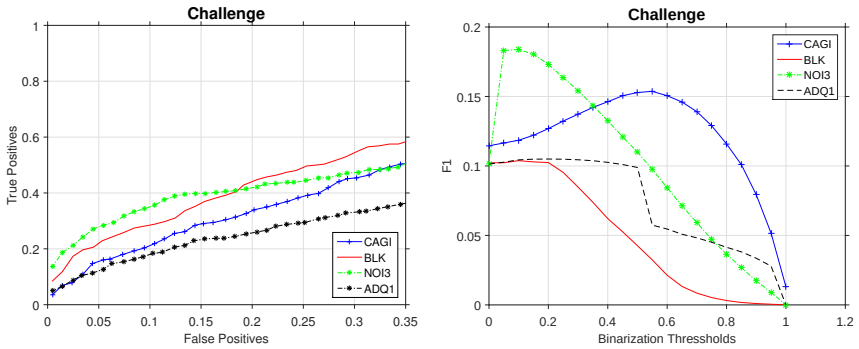
	$F1_{score} >= 0.7$		$F1_{score} >= 0.8$	
Method	**Detections**	**Unique**	**Detections**	**Unique**
ADQ1	1810	246	1561	342
BLK	578	29	392	20
CFA1	158	1	133	3
DCT	1114	6	820	7
CAGI	1711	433	1264	279
inv-CAGI	222	0	20	0
NOI1	84	0	48	0
NOI2	21	0	7	0
NOI3	1112	259	849	201

almost binarized outputs by design, is an indicative example of good readability. In the Fontani et al. dataset, ADQ1 manages to achieve good localization (mostly due to the very high performances in Classes 2 and 4) making it the best performing approach in the dataset. CAGI is a close second in terms of readability. On the other hand, BLK, which was the most competitive method in the previous evaluation, has significantly lower F1 scores.

Table 4 reports the best localized detections achieved per method. The detection threshold was set to 0.7 which generally signifies a good localization and 0.8 which is a near perfect score for most applications. The search was performed for the best binarization step for each method. *Unique* corresponds to the number of detections exclusively achieved by that method for the given F1 threshold. ADQ1 has the greatest contribution in this dataset in terms of detection, followed by CAGI and also DCT and NOI3. Concerning the unique cases, ADQ1 is outperformed by CAGI and NOI3 for the relaxed threshold and followed closely for the near perfect localizations, indicating that all three methods could be utilized in a fusion scheme not only to reinforce the detections' confidence but also in a complementary fashion. DCT on the other hand, or even BLK, manage to achieve good detections but do not contribute with unique cases because their detections are a subset of other method (i.e., DCT's detections are mostly common with ADQ1, and BLK's with CAGI, NOI3 and ADQ1)

5.2. Results on the First IFS-Challenge dataset

The Challenge dataset, being the first attempt to produce a realistic benchmark, is much harder to tackle by any single method. The performance



(a) (b)

Figure 11: Results on the Challenge dataset: a) ROC curves, b) F1 score curves.

Table 5: Reported detections on IFS-Challenge dataset for $F1_{score} >= 0.7$ and $F1_{score} >= 0.8$ at each method's best binarization threshold.

	$F1_{score} >= 0.7$		$F1_{score} >= 0.8$	
Method	**Detections**	**Unique**	**Detections**	**Unique**
ADQ1	4	1	2	0
BLK	8	0	6	2
CFA1	2	0	1	0
DCT	5	1	1	0
CAGI	16	6	9	2
inv-CAGI	3	0	2	0
NOI1	7	1	4	1
NOI2	3	1	2	1
NOI3	38	28	26	18

evaluations presented in Figure 11(a) are indicative of the above statement; there are few detections for most algorithms at a 0% false positive rate, and even when relaxing the threshold, the true positive detection rate increases slowly. Thus, any contribution in terms of unique detections and/or readable outputs is of great importance in this dataset.

Figure 11(b) presents the calculated mean F1 scores on this dataset for all competing methods. Again, in comparison with the rest of the methods, CAGI reports one of the highest F1 scores as well as readability quality as it achieves high F1 scores over a wide range of thresholds.

Table 5 reports the best localized detections achieved per method. As before, the detection thresholds were set to 0.7 and 0.8 and the search was performed for the best binarization step for each method. NOI3 has the

greatest contribution in this dataset, followed by CAGI and BLK.

5.3. Results on the Wild Web dataset

As the Wild Web dataset does not contain untampered images, the evaluations can only be performed based on the pixel-level localization accuracy on the tampered images.

Figure 12 reports the mean F1 scores calculated over the whole collection (10,646 images). Even though the values of F1 are very low for all methods one should take into account the fact that the collection consists entirely of actual forgeries sourced from the Web. The dataset is organized into 78 cases of confirmed forgery. For each case, reverse-image search engines (Google and TinEye) were used to collect as many near-duplicate instances as possible from the Web. This means that the number of instances for each case varies. Some cases have as little as 2 instances, while others more than 700. When a case that has many instances is not detectable by a method, it severely affects the calculated F1 score. Thus, the F1 curves were created by first averaging the F1 scores per case so as to minimize the impact that the unbalanced cases introduce to the calculations. CAGI presents the highest reported F1 score followed closely by NOI3. CAGI, however, additionally presents stable high F1 results for a wider range of binarization thresholds.

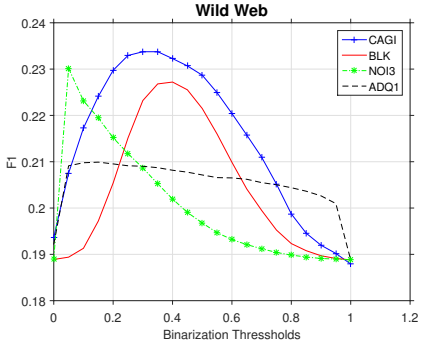


Figure 12: F1 score curves on the Wild Web dataset.

The performance of methods in the Wild Web set is also evaluated in terms of achieved detections and contribution with unique cases. As in [35] a detection is classified as correct when at least for one instance of a given sub-case the method produces an F1 score higher than a set threshold. Table 6 reports the correctly localized case detections for $F1_{score} >= 0.7$ and $F1_{score} >= 0.8$. *Detections* corresponds to the number of cases detected by

Table 6: Reported detections (total and unique) on Wild Web dataset for $F1_{score} >= 0.7$ and $F1_{score} >= 0.8$ for each method's best binarization threshold per case.

	$F1_{score} >= 0.7$		$F1_{score} >= 0.8$	
Method	**Detections**	**Unique**	**Detections**	**Unique**
ADQ1	8	1	3	0
BLK	7	0	4	0
CFA1	5	0	1	0
DCT	10	0	5	2
CAGI	19	4	6	0
inv-CAGI	22	3	13	7
NOI1	12	1	4	1
NOI2	6	0	2	0
NOI3	15	1	6	1
PENS	**33**		**21**	

the respective method, *Unique* corresponds to the number of cases detected exclusively by that method, and *PENS* (Perfect ENsemble Sum) corresponds to a theoretical perfect ensemble, where at least one method achieved detection (i.e. essentially summing the total number of cases detected out of the initial 78).

The contribution of overall detections as well as unique detections for the CAGI method (and its variant inv-CAGI) is clearly highlighted by the results. Moreover, the results indicate that the detections (i.e. F1 scores exceeding the threshold) remain prominent for a wider range of thresholds compared to competing methods. This means that in the output maps of CAGI, the value difference between the tampered and the untampered area is greater, making the visual output more striking and easy to interpret by non-experts.

5.4. Analysis of robustness and overall performance

Following the evaluations of the previous sections, we proceed to investigate the robustness of the methods when images are subjected to common post-processing operations. To this end we conducted evaluations on all metrics using the Fontani et al. and the Challenge datasets while i) resaving the images at JPEG QF90 and QF70, and ii) rescaling the images at 95%, 75% and 50% of their original size and resaving them losslessly. For brevity, when analysing the tampering detection robustness we do not present the KS curves for each algorithm, but instead estimate the threshold value for which the algorithm returns a true negative rate of 95%, and calculate the

percentage of true positives for the same threshold. As for the localization robustness, again we compactly present the results by reporting the best F1 score (at the best binarization step), per method. Figures 13 and 14 summarize the results on the Fontani et al. and Challenge dataset variants, respectively.

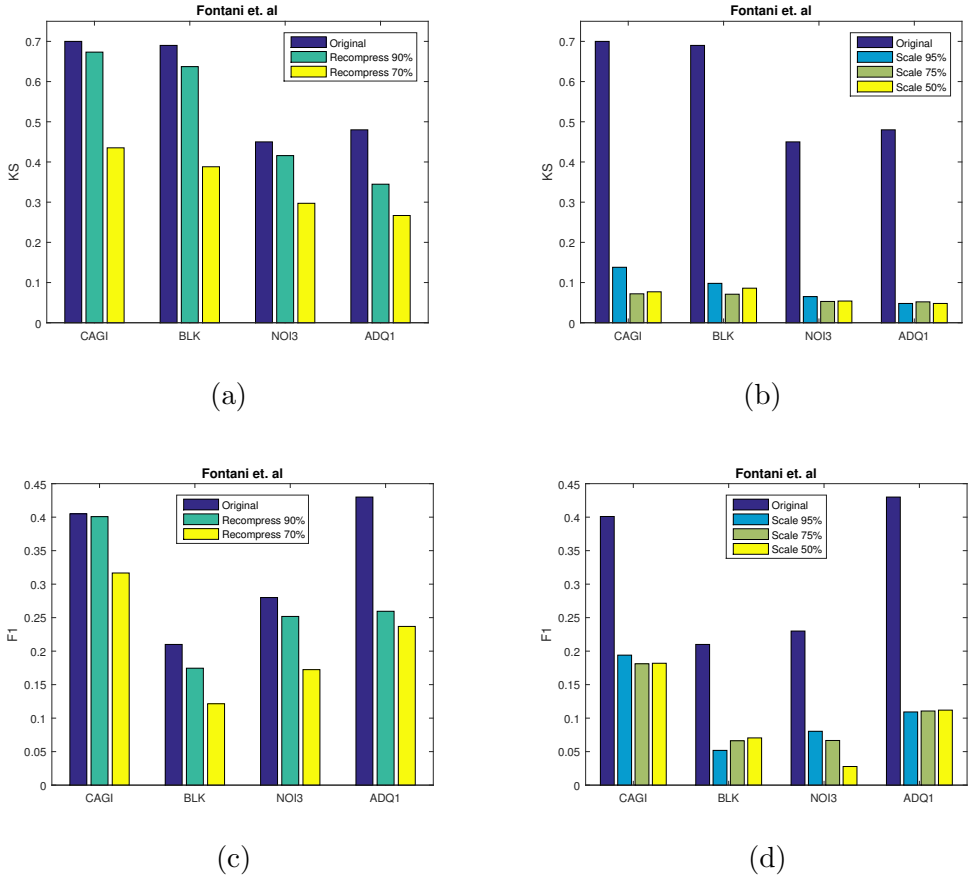


Figure 13: Evaluation of robustness on the Fontani et al. dataset: a-b) KS scores on the original dataset and after recompressions -rescales, c-d) F1 scores on the original dataset and after recompressions -rescales.

Based on the results presented in Figures 13 and 14 it is apparent that resaves (plots (a) and (c) in both figures) will cause a degree of feature degradation due to rounding errors for all methods. Both detection and localization are affected as the JPEG compression quality drops but algorithms seem to retain their performance relatively well. CAGI in particular presents

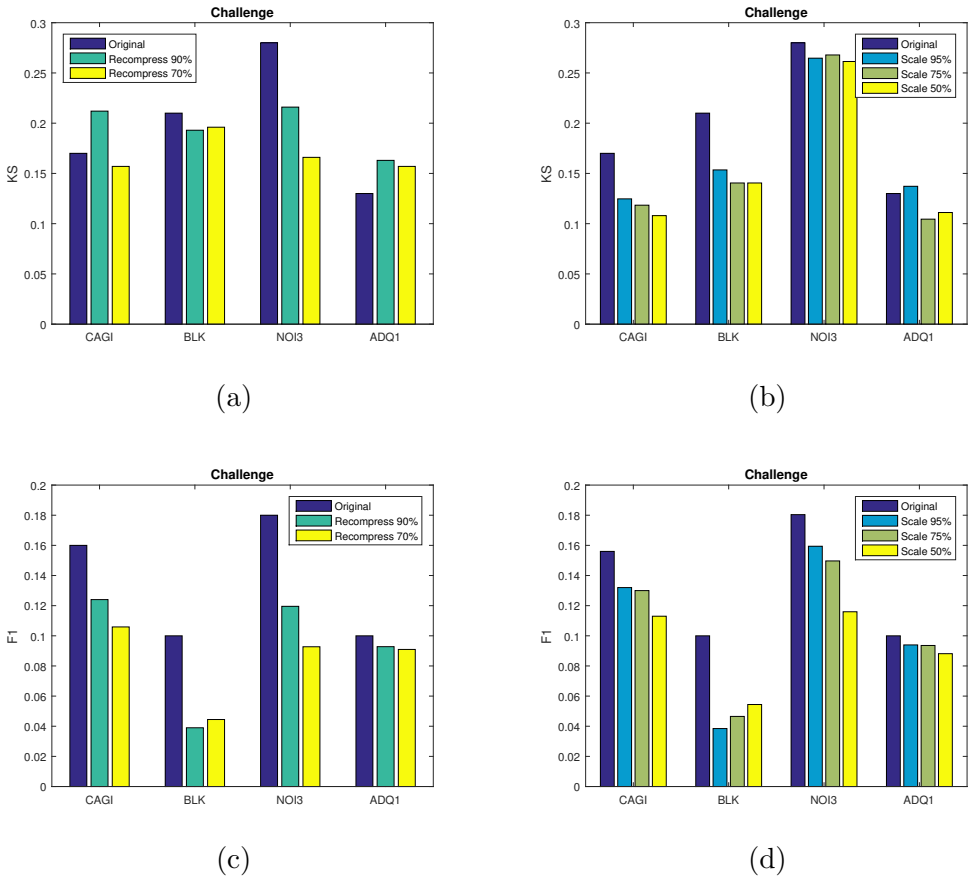


Figure 14: Evaluation of robustness on the Challenge. dataset: a-b) KS scores on the original dataset and after recompressions -rescales, c-d) F1 scores on the original dataset and after recompressions -rescales..

very robust results compared to the rest of the methods when the images have been subject to resaves. On the Fontani et al. dataset it manages to surpass the front-runners (i.e, BLK, for KS and ADQ1 for F1) and seems to be less affected by the transformation on the Challenge dataset. On the other hand rescaling operations have a much stronger impact on the detection and localization performances for all methods (plots (b) and (d) in both figures) . The effect is especially apparent in the Fontani et al. tests where the original performances were high, and less so in the Challenge sets were performances where modest to begin with. For the KS metric in particular, it should be noted that since success rates in the more complex Challenge dataset are at

Table 7: Reported detections (total) for $F1_{score} >= 0.7$ on the original datasets and after images were subjected to post-processing operations

	Fontani et al.						**Challenge**					
	Original	Recompress		Rescale			Original	Recompress		Rescale		
		90%	**70%**	**95%**	**75%**	**50%**		**90%**	**70%**	**95%**	**75%**	**50%**
CAGI	1711	1616	918	51	19	17	16	13	7	13	7	7
BLK	578	335	102	0	0	0	8	5	6	3	3	2
NOI3	1112	946	551	15	6	1	38	21	13	34	31	26
ADQ1	1810	989	629	0	0	0	4	3	2	2	2	2

best around 20%, which, given the difference between the masks used for positive detection and negative detection may be rather close to random, analysis on the detection robustness can unfortunately not be reliable.

To better focus on the performance degradation caused by these transformations, we decided to study only those images that were correctly detected on the original version (i.e. had tampering localizations of $F1_{score} >= 0.7$ prior to the transformations). Table 7 shows the number of successful detections on these particular images after they were subjected to transformations, compared to the detections achieved originally. As can be seen, CAGI is very robust with respect to recompressions. On the other hand, rescaling has a clear negative impact for all methods and datasets. An exception could be the case of NOI3 and CAGI for the Challenge dataset, but the limited number of original detections does not allow us to draw broader conclusions.

Moving on the discussion to the overall performance, Figure 15 summarizes the recorded performance of all methods on the three employed evaluation criteria; i) the ability of a method to retrieve true positives of tampered images at a low level of false positives (KS@0.05); ii) the ability to achieve good localization of the tampered region within the image (F1), and; iii) the readability of the produced heat map, i.e. a high distinction of assigned values for pixels belonging to tampered versus untampered regions, expressed as the range of different binarization thresholds that result in high F1 scores ($> 70\%$ of the respective maximum F1 score).

At this point some overall remarks concerning the two last criteria should be made: The proposed method is not only performing among the top methods concerning the F1 score in all three datasets, but has also a good (wide) range of possible binarization thresholds that lead to high F-score for all tested datasets. This attribute is of great importance since it could be leveraged within an automated binarization process. For instance, CAGI can be expected to produce close to optimum F1 score (localization) for a threshold

	Fontani et. al			Challenge			Wild Web	
	KS	F1	[thress range]	KS	F1	[thress range]	F1	[thress range]
CAGI	0.70	0.40	**0,3-0,7**	0.17	0.16	**0-0,8**	0.234	0,15-0,6
inv-CAGI	0.31	0.19	0,8-0,95	0.08	0.11	**0-0,95**	0.272	**0,4-0,95**
BLK	0.69	0.21	0,35-0,65	0.21	0.10	0-0,35	0.227	0,3-0,5
NOI3	0.45	0.28	0,05-0,4	0.28	0.18	0,05-0,4	0.230	0,05-0.1
ADQ1	0.48	0.43	**0,05-0,95**	0.13	0.10	0-0,5	0.209	**0,05-0,8**
DCT	0.53	0.33	0,25-0,55	0.25	0.11	0-0,65	0.246	0,3-0,6
NOI1	0.23	0.12	0-0,35	0.21	0.09	0-0,25	0.249	0,1-0,35
NOI2	0.08	0.12	0-0,3	0.04	0.10	0-0,05	0.225	0,05-0,2
CFA1	0.05	0.13	0-0,25	0.01	0.10	0-0,2	0.203	0,1-0,3

Figure 15: Summary of results: performance of methods in the three datasets: KS score (retrieved True Positives for 5% False Positives rate), F1 score (maximum reported value); thres. range (binarization step range that produces F1 scores $> 70\%$ of the respective maximum value)

of 0.4 or 0.5 and inv-CAGI for higher values of 0.8 and above. Other methods, e.g. BLK, have ranges that fall into completely different values in the three datasets, or have limited binarization levels to choose from (e.g NOI2, NOI3).

Overall, the results of Figure 15 attest the versatility of the proposed method. The method manages to maintain high performance in all three tested metrics across all datasets showcasing a good balance between detection and localization of forgeries, as well as high readability of its outputs.

Another advantage of the method is that it does not require parameter selection (since a reasonable choice for the binarization threshold works very well across all dataset). This makes it ideal for use in practical settings by non-experts. In fact, the method has been integrated in a web-based image forensics service that has been co-designed and tested in realistic settings by journalists and media experts [36].

6. Conclusion

The paper presented a novel tampering localization method based on JPEG blocking artifacts discontinuities for detecting splices. The key design goals for the method have been high robustness over a variety of forgery cases, achievement of successful detections in cases where other algorithms fail, and the generation of "clean" outputs that are easy to interpret by non-experts.

Experiments were performed on both synthetic datasets and realistic/real tampering cases, and the proposed method was directly compared to seven state-of-the-art techniques, representing different classes of forensic analysis. Experimental results across all datasets demonstrated that the method is robust in terms of localization accuracy and readability of the produced

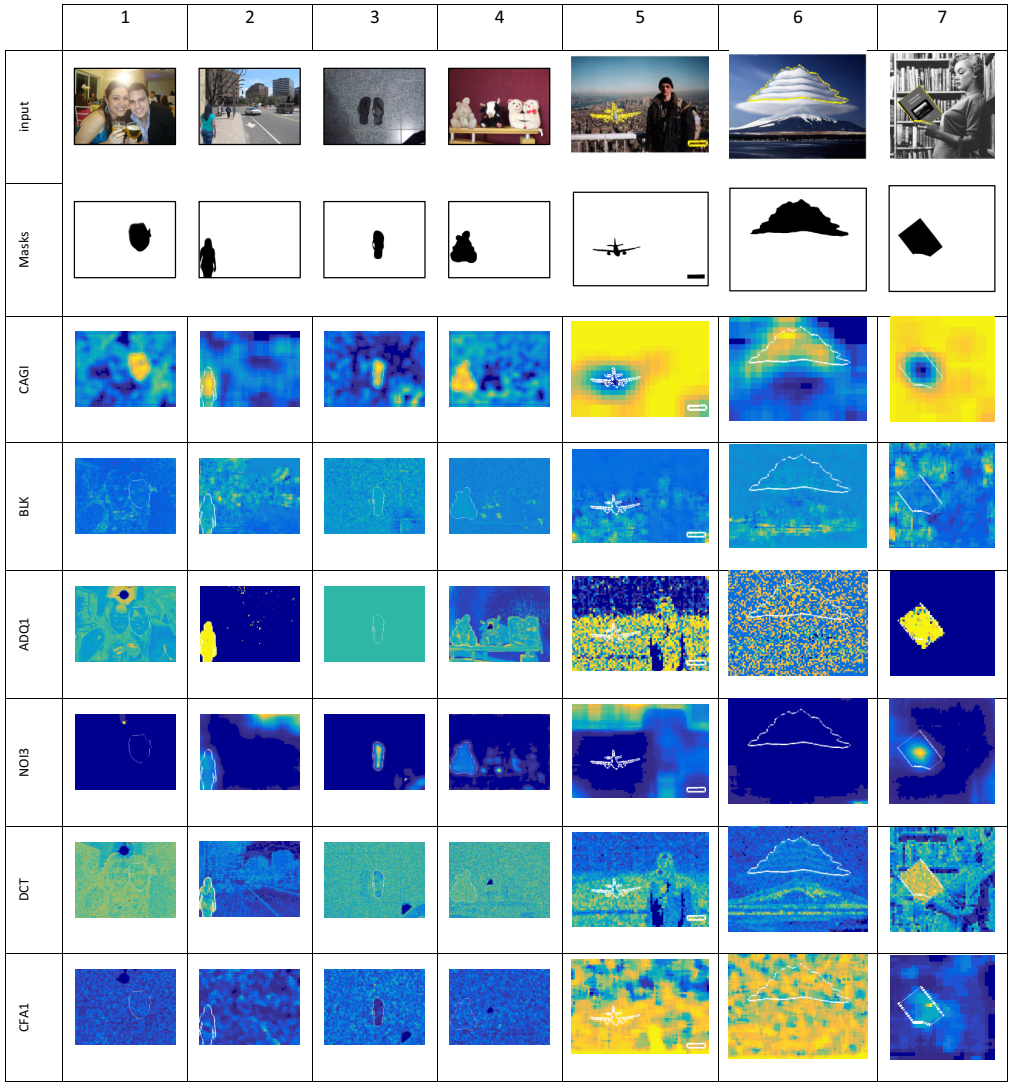


Figure 16: Heat maps produced by the methods: Input images in columns 1-4 are taken from the Challenge dataset and columns 5-7 from the Wild Web dataset. For the proposed method, the outputs shown for input images 1-4 and 6 are produced by CAGI, while for images 5 and 7 they were produced by inv-CAGI. The tampered part in each case is drawn using a white outline on all heat maps.

outputs. More importantly, since the reported detections contributed by our method during the experimental evaluation include many unique cases (i.e.

where other algorithms fail) we conclude that including it in an ensemble forensics analysis system would significantly improve its detection performance. This is a direction we plan to explore in the future.

Acknowledgment

This work has been supported by the REVEAL, InVID and TENSOR projects, partially funded by the European Commission under contract no. FP7-610928, H2020-687786 and H2020-700024 respectively.

[1] M. C. Stamm, M. Wu, K. R. Liu, Information forensics: An overview of the first decade, IEEE Access 1 (2013) 167–200.

[2] M. Zampoglou, S. Papadopoulos, Y. Kompatsiaris, A large-scale evaluation of splicing localization algorithms for web images, Multimedia Tools and Applications (2016) Online firstdoi:10.1007/s11042-016-3795-2.

[3] Z. Fan, R. L. De Queiroz, Identification of bitmap compression history: Jpeg detection and quantizer estimation, IEEE Transactions on Image Processing 12 (2) (2003) 230–235.

[4] E.-S. M. El-Alfy, M. A. Qureshi, Robust content authentication of gray and color images using lbp-dct markov-based features, Multimedia Tools and Applications (2016) Online firstdoi:10.1007/s11042-016-3855-7.

[5] X. Qiu, H. Li, W. Luo, J. Huang, A universal image forensic strategy based on steganalytic model, in: A. Unterweger, A. Uhl, S. K. 0001, R. Kwitt, A. Piva (Eds.), ACM Information Hiding and Multimedia Security Workshop, IH&MMSec '14, Salzburg, Austria, June 11-13, 2014, ACM, 2014, pp. 165–170.

[6] J. Fan, T. Chen, J. Cao, Image tampering detection using noise histogram features, in: International Conference on Digital Signal Processing (DSP), IEEE, 2015, pp. 1044–1048.
URL `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7227493`

[7] Q. Zhang, W. Lu, J. Weng, Joint image splicing detection in dct and contourlet transform domain, Journal of Visual Communication and Image Representation 40 (2016) 449–458.

[8] A. C. Popescu, H. Farid, Exposing digital forgeries by detecting traces of resampling, IEEE Transactions on signal processing 53 (2) (2005) 758–767.

[9] Y. Su, X. Jin, C. Zhang, Y. Chen, Hierarchical image resampling detection based on blind deconvolution, Journal of Visual Communication and Image Representation 48 (2017) 480–490.

[10] W. Li, Y. Yuan, N. Yu, Passive detection of doctored jpeg image via block artifact grid extraction, Signal Processing 89 (9) (2009) 1821–1829.

[11] Z. Lin, J. He, X. Tang, C.-K. Tang, Fast, automatic and fine-grained tampered jpeg image detection via dct coefficient analysis, Pattern Recognition 42 (11) (2009) 2492–2501.

[12] T. Bianchi, A. Piva, Image forgery localization via block-grained analysis of jpeg artifacts, IEEE Transactions on Information Forensics and Security 7 (3) (2012) 1003–1017.

[13] A. E. Dirik, N. D. Memon, Image tamper detection based on demosaicing artifacts., in: ICIP, 2009, pp. 1497–1500.

[14] P. Ferrara, T. Bianchi, A. De Rosa, A. Piva, Image forgery localization via fine-grained analysis of cfa artifacts, IEEE Trans. on Information Forensics and Security 7 (5) (2012) 1566–1577.

[15] M. K. Johnson, H. Farid, Metric measurements on a plane from a single image, Dept. Comput. Sci., Dartmouth College, Tech. Rep. TR2006-579.

[16] S. Gholap, P. Bora, Illuminant colour based image forensics, in: IEEE Region 10 Conference - TENCON 2008, IEEE, 2008, pp. 1–5.

[17] B. Mahdian, S. Saic, A bibliography on blind methods for identifying image forgery, Signal Processing: Image Communication 25 (6) (2010) 389–399.

[18] A. Rocha, W. Scheirer, T. Boult, S. Goldenstein, Vision of the unseen: Current trends and challenges in digital image and video forensics, ACM Computing Surveys (CSUR) 43 (4) (2011) 26.

[19] G. K. Birajdar, V. H. Mankar, Digital image forgery detection using passive techniques: A survey, Digital Investigation 10 (3) (2013) 226–245.

[20] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, S. Tubaro, Aligned and non-aligned double jpeg detection using convolutional neural networks, Journal of Visual Communication and Image Representation 49 (2017) 153–163.

[21] S. Ye, Q. Sun, E.-C. Chang, Detecting digital image forgeries by measuring inconsistencies of blocking artifact, in: 2007 IEEE International Conference on Multimedia and Expo, IEEE, 2007, pp. 12–15.

[22] T. Bianchi, A. De Rosa, A. Piva, Improved dct coefficient analysis for forgery localization in jpeg images, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2011, pp. 2444–2447.

[23] I. Amerini, R. Becarelli, R. Caldelli, A. Del Mastio, Splicing forgeries localization through the use of first digit features, in: 2014 IEEE International Workshop on Information Forensics and Security (WIFS), IEEE, 2014, pp. 143–148.

[24] H. Farid, Exposing digital forgeries from jpeg ghosts, IEEE Trans. on information forensics and security 4 (1) (2009) 154–160.

[25] B. Mahdian, S. Saic, Using noise inconsistencies for blind image forensics, Image & Vision Com. 27 (10) (2009) 1497–1503.

[26] S. Lyu, X. Pan, X. Zhang, Exposing region splicing forgeries with blind local noise estimation, International Journal of Computer Vision 110 (2) (2014) 202–221.

[27] D. Cozzolino, G. Poggi, L. Verdoliva, Splicebuster: a new blind image splicing detector, in: Information Forensics and Security (WIFS), 2015 IEEE International Workshop on, IEEE, 2015, pp. 1–6.

[28] H. Zeng, Y. Zhan, X. Kang, X. Lin, Image splicing localization using pca-based noise level estimation, Multimedia Tools and Applications 76 (4) (2017) 4783–4799.

[29] H. Yao, F. Cao, Z. Tang, J. Wang, T. Qiao, Expose noise level inconsistency incorporating the inhomogeneity scoring strategy, Multimedia Tools and Applications (2017) 1–23.

[30] D. Cozzolino, L. Verdoliva, Single-image splicing localization through autoencoder-based anomaly detection, in: Information Forensics and Security (WIFS), 2016 IEEE International Workshop on, IEEE, 2016, pp. 1–6.

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

[32] M. Fontani, T. Bianchi, A. De Rosa, A. Piva, M. Barni, A framework for decision fusion in image forensics based on dempster–shafer theory of evidence, IEEE Transactions on Information Forensics and Security 8 (4) (2013) 593–607.

[33] V. Vonikakis, A. Gasteratos, I. Andreadis, Enhancement of perceptually salient contours using a parallel artificial cortical network, Biological cybernetics 94 (3) (2006) 192–214.

[34] Report on the IEEE-IFS Challenge on image forensics, `http://ifc.recod.ic.unicamp.br/fc.website/index.py`, accessed: 20-03-2016.

[35] M. Zampoglou, S. Papadopoulos, Y. Kompatsiaris, Detecting image splicing in the wild (web), in: IEEE International Conference on Multimedia & Expo Workshops (ICMEW), IEEE, 2015, pp. 1–6.

[36] M. Zampoglou, S. Papadopoulos, Y. Kompatsiaris, R. Bouwmeester, J. Spangenberg, Web and social media image forensics for news professionals., in: SMN@ ICWSM, 2016.