Forgery Detection by Internal Positional Learning of Demosaicing Traces

Quentin Bammey, Rafael Grompone von Gioi, Jean-Michel Morel
Centre Borelli – École Normale Supérieure Paris-Saclay – Université Paris-Saclay

Abstract

We propose 4Point (Forensics with Positional Internal Training), an unsupervised neural network trained to assess the consistency of the image colour mosaic to find forgeries. Positional learning trains the model to learn the modulo-2 position of pixels, leveraging the translation-invariance of CNN to replicate the underlying mosaic and its potential inconsistencies. Internal learning on a single potentially forged image improves adaption and robustness to varied post-processing and counter-forensics measures. This solution beats existing mosaic detection methods, is more robust to various post-processing and counter-forensic artefacts such as JPEG compression, and can exploit traces to which state-of-the-art generic neural networks are blind. Check qbammey.github.io/4point for the code.

1. Introduction

Image forgeries are omnipresent, from fake news in social networks [53] to scientific misconduct. Indeed, a large variety of image processing tools are available to create visually realistic image alterations. Image forensics aims at detecting and characterizing these alterations. This can be done by analysing traces left behind by the image processing chain, and detecting its local inconsistencies caused by tampering. This analysis is difficult because of the variety of interactions between the successive image processing steps, such as demosaicing, gamma correction, and JPEG encoding. These operations are often complemented by further, equally unknown, post-processing such as resampling and even potential anti-forensic attacks.

The large field of image forensics has been approached in two ways: syntactic analysis and deep learning. The recent deep learning models are directly or indirectly trained to find forgeries in images. ManTraNet [59] is an end-to-end network trained directly on forged images to extract their features and localize inconsistent regions. Noiseprint [14] extracts the noise pattern of the image and finds its inconsistencies by means of Siamese networks.

Bappy et al. [6, 7] approach forgery detection as a binary segmentation problem between forged and authentic regions. The Achilles’ heel of these methods is the training database itself. Working very well on images and forgeries similar in nature to the database, they may nevertheless fail to adapt to the variety of images and forgery methods.

Syntactic analysis instead focuses on specific inconsistencies left by forgeries. For instance, Error Level Analysis [28] compresses the image and uses the residual between the original and compressed versions to find areas in which compression levels are different. Many methods look for inconsistencies in the traces left by JPEG compression [47, 58, 27, 39, 40, 9, 1] or noise discrepancies [30, 51, 13, 22, 43, 44, 8, 45, 35, 56, 29, 15, 48, 42, 41, 60, 57]. However, forgery tools are varied, and so is the plethora of...
postprocessing operations, both occurring naturally through the treatment and distribution of the image, or maliciously made to hide the forgery. The syntactic methods therefore often miss forgeries that did not modify the statistics of the image in an expected way, or whose traces were erased by further processing. Nevertheless, these methods yield explainable and easily understood results.

One of the classic syntactic methods is to analyse the traces of demosaicing. Most cameras recreate the colour image from a Bayer pattern (shown in Fig. 2a) that samples just one colour channel per pixel. This pattern is $2 \times 2$ periodic. Demosaicing is a clever nonlinear interpolation method recreating the two missing colour channels at each pixel from the observed channel values in a neighborhood. Demosaicing algorithms have shown in recent year a growing sophistication, from standard manual methods [26, 49, 20, 21] to complex classic ones [31, 32, 46] and eventually to deep-learning-based [55].

Demosaicing is of interest to image forensics, because when an image is forged by splicing or copy-move, there is a $3/4$ chance that the mosaic patterns of the authentic and forged regions will not be aligned. Being able to detect local inconsistencies of the original mosaic pattern can thus help find such forgeries. Most manual methods [4, 12, 54] easily detect the presence of demosaicing, but fail to detect the position of the mosaic pattern, due to the extreme variety of demosaicing algorithms. Recently, Bammey et al. [3] used deep learning to identify the correct position, even on unseen demosaicing algorithms. However, even recent state-of-the-art methods fail to detect the mosaic pattern of images after common post-processing such as JPEG compression, even at a very high quality factor. Furthermore, post-processing, and especially JPEG compression, can be done in various ways. As a consequence, deep learning-based methods must take into account the impossibility to learn on all existing demosaicing algorithms.

In this article, inspired by [3, 38], we propose a self-supervised learning method that leverages the translation invariance of a CNN to implicitly interpret demosaicing traces. Inconsistencies in the detected traces are then evidence of possible forgeries (see Fig. 1). We borrow from [3] the idea of performing self-supervised learning of the demosaicing grid position. Our method enables internal learning on a single test image, without the need for a large database of similarly-processed images. Using our own JPEG compression in several different alignments further improves robustness to JPEG compression. Furthermore, it becomes able to detect not only regions with a different grid position, but also regions with incoherent grid positions, like those caused by most inpainting methods. The method detects separately the diagonal inconsistencies, that are generally more solid, then detects the general pattern inconsistencies. Last but not least, we process the results of the network to filter out irrelevant results, instead of relying on the often-overconfident [25] raw results of the neural network. Overall, our results are more robust and interpretable.

We show that the CNN can be directly fine-tuned on a single image (internal learning) to assess its consistency, by training on it as if it were authentic, and that this retraining makes the network more robust to various post-processing methods. As a consequence, our network is able to adapt to a broader range of images, even if the network was not initially trained on the same specific post-processing chain.

Because JPEG compression destroys the high frequencies of an image in which demosaicing traces are located, there is no hope to detect them when an image has suffered strong compression or has been downsampled. This means that demosaicing artefact detection is irrelevant to images found, for instance, in social media. Nevertheless, being able to achieve some degree of robustness when the JPEG compression is not too strong can make such methods useful in other many areas, such as criminal investigations, press authentication or photographic contest verification.

To tackle the variety of existing post-processing algorithms and combinations thereof, we show that the network can be fine-tuned on-the-fly on a single, potentially forged, image by training on it as if it was authentic. Adapting to the image’s statistics in this way helps the network improve its interpretation of demosaicing traces over common post-processing methods and detect forgeries.

### 2. Related Works

In a pioneer paper [50], Popescu and Farid proposed to use an expectation-maximization (EM) algorithm to jointly estimate the demosaicing algorithm of an image and detect the local Bayer pattern. A Fourier transform of the resulting pseudo-probability map then shows the presence or absence of demosaicing on the image. The methods in [24, 2] replace the EM algorithm with a direct linear estimation of the algorithm in all four possible positions. The method of [24] further uses the Discrete Fourier Transform (DCT) instead of the FFT, to detect not just the mere presence of demosaicing, but also changes of the mosaic position. Indeed, a change in the sign of the DCT is easier to detect than a phase change in the FFT. All of these methods assume
demosaicing was performed independently in each colour channel and a linear estimation is a valid representation of the demosaicing process. These assumptions are no longer true with most commonly-used demosaicing methods.

Kirchner [34] proposed to directly detect the mosaic position used in the image. To do so, the authors mosaic the image in all four possible positions, and redemosaic it with bilinear interpolation. When this is done in the correct position, the processed image is closer to the original image, enabling detection of the correct pattern. Due to the large variety of demosaicing algorithms, this method does not work well in the wild, but can provide reliable results when the algorithm used by the camera is known or can be estimated.

Choi et al. [12] remarked that interpolated pixels are more likely to take intermediate values than sampled pixels. Counting the number of intermediate values in each of the four patterns thus provides a simple estimate the pattern, although the decision can be confused with modern algorithms that do not process colour channels independently.

Shin [54] uses the fact that most algorithms do not process colour channels, but more or less directly the difference between the green and the other two channels. As a consequence, they work on the difference of channels to find the pattern with the highest variance, which corresponds to a higher probability of being originally sampled, thus breaking free from the assumption of channel independence.

In previous work [3], we trained a self-supervised neural network that detects the relative position modulo 2 of pixels and blocks in the image. Because a CNN is invariant to translation, it implicitly needs to learn from mosaic artefacts in order to perform this classification. Shifts in the output of the network are thus evidence of forged regions. This network can be fine-tuned on a dataset of potentially forged images, which enables it to adapt to it and gain some robustness to JPEG compression.

In recent years, internal learning has gained popularity in several domains of image processing. Using self-supervised information to retrain an over-parameterized network on the very image to process, such methods often provide better results and greater adaptability to uncontrolled cases than their supervised counterparts. For instance, in the close field of image denoising, Noise2Void [38] and the subsequent Self2Self [52] train a network to reconstruct a noisy image while hiding pixels from its input. The network provides the regularization necessary to actually denoise the image.

3. Method

We propose to train a fully-convolutional neural network (CNN) to detect the positional information on each pixel of an image. Because a CNN is invariant to translation, it does not know these positions and has to infer them from camera traces, in particular from demosaicing traces. Inconsistencies in the output of the network reveal inconsistencies in the demosaicing pattern, and are thus traces of forgeries. In Sec. 3.1, we explain how the network is trained on positional data. In Sec. 3.2, we show how some level of robustness to JPEG compression can be achieved during training. In Sec. 3.3, we show that it is possible to retrain this network directly on a single suspicious image to adapt to the setup of this image, thus achieving a greater robustness to JPEG compression. Finally, in Sec. 3.4, we detail how the network’s output can be used to detect forgeries.

3.1. Positional learning

A CNN is invariant to translation; it does not know the position of an image’s pixels. If we train it to do so, it thus has to rely on external cues, such as demosaicing traces. After such training, the network is thus able to provide an insight as to the image’s demosaicing.

Bammey et al. [3] proposed to train a CNN to detect the modulo 2 position of each pixel, both horizontally and vertically. Coarser-scale position information is irrelevant to the very local demosaicing traces, and could lead the network to rely on unwanted cues. However, demosaicing detection algorithms [4, 12, 54, 34] usually proceed in two steps: they start by detecting the diagonal pattern, i.e. to find which pixels were sampled in green, then try to distinguish between the two patterns sharing the same diagonal, because the diagonal is easier and more robust than directly making a decision on the full pattern.

To adopt this, we trained a CNN to detect the offset of each pixel’s diagonal (Fig. 4a), representing whether the pixel is sampled in green. For pixels that are not sampled in green, we also estimate whether they are on an even line (Fig. 4b), saying whether they are sampled in red or blue.

This method is self-supervised, since the position of each pixel is known. The only requirement is that all images of the training set be demosaiced in the R G G B pattern, to enforce consistency of the output across images and correspondence between the detected position and the sampled colour. If a training image has been demosaiced in another pattern, we align it to the R G G B by cropping its first row and/or column.

We use a 17 × 64 DnCNN [61] architecture, with 17 layers and 64 features per layer. DnCNN is more suited to our

![Figure 3: The four possible sampling patterns can be grouped by the diagonal on which the green channel was sampled: R G G B and G B G R share the diagonal, whereas G B R G and R G B G share the other one.](image-url)
task than other standard structures, as it does not make use of any downsampling. The downsampling which is found in most CNN structures would remove the high-frequency information in which demosaicing artefacts are located.

3.2. Training on JPEG-compressed images

JPEG compression is a major obstacle to demosaicing detection. The quantization induced during compression quickly removes the highest frequencies of an image, in which demosaicing traces are located. As a consequence, strongly-compressed images keep no demosaicing traces and cannot be detected by our method. Nevertheless, it remains possible to find demosaicing traces on high-quality images, where the compression is minimal. This prompts us to train our network on JPEG-compressed images.

However, JPEG compression aggregates data in $8 \times 8$ blocks, imprinting a strong 8-periodic component. JPEG artefacts would thus be enough for the CNN to find the position of all pixels without even looking at demosaicing traces. To prevent the network from doing this and to force it to analyse the demosaicing artefacts over JPEG compression, we start from uncompressed images and compress them ourselves in the four possible alignments between the demosaicing pattern and the JPEG grid, seen in Fig. 5. For each initial image, we have 4 compressed images with different shifts, and the network is trained simultaneously on them. The network can no longer directly analyse and use JPEG compression to find the positions, as the positional cues from JPEG contradict each other across the shifts.

This training scheme assumes that the JPEG grid is consistent across the image, i.e., that the compression happened after the forgery. This assumption is reasonable; if the JPEG grid itself is inconsistent, the forgery will be apparent and much easier to detect through JPEG grid analysis [47, 5].

3.3. Internal learning

Training on compressed images already provides some robustness to JPEG compression. However, there is a huge variety of post-processing algorithms, and an even bigger number of combinations thereof, as post-processing encompasses not only algorithms such as JPEG compression, but also image-enhancing filters often automatically applied by cameras, and specific counter-forensic measures such as added noise or median filtering. Furthermore, such algorithms change with time. We propose an alternative strategy. The robustness of our network can be further increased with internal learning, i.e. by retraining the network on the specific image we want to study. This enables the network to adapt to the specific statistics of the image.

Given a potentially forged image to analyse, we assume the image is authentic and train the network repeatedly on it to detect the position of pixels, as explained in Sec. 3.1. Contrarily to the initial training, the image is not necessarily in the $G\ B\ R\ G\ B\ R\ G\ B\ R\ G\ B\ R\ G\ B\$. To train the network, we compute the loss not only on the initial target of Fig. 4, but we also shift the target by one row and/or column. We thus have 4 losses depending on the offset of the target, we use the one that is minimal across the global image. In other words, we train the network to be locally coherent with the global pattern.

Of course, single-image training induces a risk of overfitting. However, even if the network overfits on the image, training is done on the hypothesis that the full image is authentic. In other words, fine-tuning incentivises the network to conclude that everything is authentic. As a consequence, if the image is actually authentic, the risks of making a false detection are lower than with the pretrained-network; even overfitting will by design not induce new false detections. In the more interesting case where the image is indeed forged, the network will also adapt to the post-processing to learn demosaicing traces and detect the position of pixels. Forged regions in images are usually small compared to the total size. As a consequence, even though the forged regions would steer the network towards detecting their pixels’ positions correctly, they would produce evidence contradicting that of much larger authentic regions, and the network should thus not learn too much from forged regions. The small size and locality of the network is particularly impor-

![Figure 4: The CNN is trained to output these two patterns. Asterisks (*) correspond to values that are ignored.](image)

(a) The first detected feature is the diagonal offset. It corresponds to whether the pixel was sampled in green. 
(b) The second feature is the evenness of the line/column of spords to whether the pixel was sampled in red or in blue.

![Figure 5: The JPEG grid and the Bayer pattern can be aligned in four different ways. Simultaneous training of the network on the four possible shifts prevents it from using JPEG information instead of the mosaic.](image)
tant here to prevent it from being able to adapt to both, the authentic and the forged regions. The impact of overfitting can thus be expected to be limited.

3.4. Forgery detection

The network does not directly detect forgeries in an image, it only detects pixel-wise, demosaicing-related positional information. This information must then be analysed to find forged regions of an image.

The output of the network consists of two feature maps, following the targets of Fig. 4: the diagonal of the pixel $O_d$, and the line $O_l$ for pixels on the main diagonal.

These results are aggregated in $2 \times 2$ blocks, corresponding to a Bayer CFA tile from Fig. 2a. Let $B_d$ and $B_l$ represent a block from $O_d$ and $O_l$, three binary decisions are made on that block:

- $\delta \cdot G \cdot G \cdot G \cdot G \triangleq \lfloor \frac{1}{2} (B_{d_{0,0}} + B_{d_{1,1}} - B_{d_{0,1}} - B_{d_{1,0}}) \rfloor$, 
- $\delta \cdot G \cdot B \cdot G \cdot G \triangleq \lfloor \frac{1}{2} (B_{l_{0,0}} - B_{l_{1,1}}) \rfloor$, and
- $\delta \cdot G \cdot B \cdot B \cdot G \triangleq \lfloor \frac{1}{2} (B_{l_{1,0}} - B_{l_{1,0}}) \rfloor$.

In these definitions, $\lfloor \cdot \rfloor$ represents rounding to 0 or 1. $\delta \cdot G \cdot G \cdot G \cdot G$ says in which diagonal the block – or rather its top-left pixel – is detected, from which the pattern diagonal can be inferred: a value of 0 (resp. 1) means the block is demosaiced in a $\cdot G \cdot G \cdot G \cdot G$ (resp. $\cdot G \cdot G \cdot G \cdot G$) pattern. Assuming we know the block is demosaiced in one of the two $\cdot G \cdot G \cdot G \cdot G$ patterns, $\delta \cdot G \cdot B \cdot G \cdot G$ then distinguishes them; a value of 0 (resp. 1) means the block is demosaiced in the $\cdot G \cdot G \cdot G \cdot G$ (resp. $\cdot G \cdot G \cdot G \cdot G$) pattern. Similarly, a value of 0 (resp. 1) for $\delta \cdot G \cdot B \cdot B \cdot G$ means the block is demosaiced in the $\cdot G \cdot G \cdot G \cdot G$ (resp. $\cdot G \cdot G \cdot G \cdot G$) pattern, assuming we know the block is demosaiced in a $\cdot G \cdot G \cdot G \cdot G$ pattern.

Using these blocks, we determine the pattern of the global image. The main diagonal pattern $G_D$ is the mode of $\delta \cdot G \cdot G \cdot G \cdot G$ across the whole image ($\cdot G \cdot G \cdot G \cdot G$ or $\cdot G \cdot G \cdot G \cdot G$). The full pattern $G_P$ is then the mode of either $\delta \cdot G \cdot B \cdot G \cdot G$ or $\delta \cdot G \cdot B \cdot B \cdot G$.

At this stage, we can already have a first estimation of inconsistent regions in an image, by considering a block forged if its detected pattern is different from that of the full image. However, such results are not stable enough. We instead propose to aggregate the blocks in overlapping windows of $W \times W$ blocks, or $2W \times 2W$ pixels. For each window, we define $\Delta \cdot G \cdot G \cdot G \cdot G$ (resp. $\Delta \cdot G \cdot B \cdot G \cdot G$, $\Delta \cdot G \cdot B \cdot B \cdot G$) as the mean of $\delta \cdot G \cdot G \cdot G \cdot G$ (resp. $\delta \cdot G \cdot B \cdot G \cdot G$, $\delta \cdot G \cdot B \cdot B \cdot G$) across the window. The detected diagonal pattern of the window is

$$D \triangleq \begin{cases} \cdot G \cdot G \cdot G \cdot G & \Delta \cdot G \cdot G \cdot G \cdot G < \frac{1}{2}, \\ \cdot G \cdot G \cdot G \cdot G & \text{otherwise} \end{cases}$$

the demosaicing pattern is then detected as:

$$P \triangleq \begin{cases} R & G \quad \text{and} \quad \Delta \cdot G \cdot G \cdot G \cdot G \leq \frac{1}{2} \\ B & G \quad \text{and} \quad \Delta \cdot G \cdot G \cdot G \cdot G > \frac{1}{2} \\ R \quad \text{and} \quad \Delta \cdot G \cdot G \cdot G \cdot G \leq \frac{1}{2} \\ B \quad \text{and} \quad \Delta \cdot G \cdot G \cdot G \cdot G > \frac{1}{2} \end{cases}$$

We can now introduce a forgery detection confidence score to windows. Windows that are detected in the same grid than the original image, i.e. where $P = G_P$, are given a zero confidence; they are considered authentic. Windows whose detected diagonal is different than the original image’s, i.e. where $D \neq G_D$, are given an initial confidence of $2 \cdot \Delta \cdot G \cdot G \cdot G \cdot G - \frac{1}{2}$. If a region is detected with high confidence as forged, nearby regions that may be detected with a lower confidence are likely to be part of the forgery as well. Consequently, we segment the detected windows into connected components. Two inconsistent windows are connected if they can be joined by a path of windows of the same diagonal. The detection confidence of all the windows in this component is then set to the value of the most confident of those windows.

Likewise, windows whose detected diagonal is consistent, but whose pattern is not, i.e. where $D = G_D$ but $P \neq G_P$, are given an initial confidence of $2 \cdot \Delta \cdot G \cdot G \cdot G \cdot G$ depending on the diagonal. Segmentation in connected components of the same grid and component-wise maximum are applied as above.

4. Experiments

4.1. Experiments setup

To train our network, we used the 1488 raw images from the Dresden database [23] and demosaiced them with several demosaicing methods: AICC [16, 17], R1 [31], MLR1 [32], AR1 [46], CDMDCNN [55, 18], CS [21], GBTF [49], Alternating Projections [19], HA [26], LMMSE [20] and bilinear demosaicing. Training was done with Adam [33] with a learning rate of $10^{-3}$. During internal training on each image, the network was retrained for 15 iterations.

Experiments were done on two datasets. The Korus [36, 37] database contains 220 high-quality splicing forgeries, on images from 4 different cameras. The Trace [5] database introduces invisible forgery traces in the form of pipeline inconsistencies. Each raw image is processed with two different camera pipelines, then both obtained images are merged according to a forgery mask. The content of the image stays unchanged, only the pipeline is altered. The database is divided into several datasets with the same images and masks but different changes in the pipeline. Two of these are of particular interest to us; the CFA grid dataset, in which the forged region is demosaiced in a different pattern; and the
CFA algorithm, in which the forged region is demosaiced with a different algorithm, and a new demosaicing pattern is selected, which thus has a \( \frac{3}{4} \) chance to be different from the authentic region. For each image and dataset, two forgeries are available, using the same pipeline but different masks, one taken from the image segmentation (endo-mask) and the other from another image (exo-mask). As our method only detects pattern shifts, we cannot expect to make detections in regions where the demosaicing pattern is the same, i.e. in one quarter of the images of the CFA algorithm dataset. However, since our method is not content-aware, results between the endo-mask and exo-mask should be similar. On the CFA grid dataset, we further study our robustness to JPEG compression, by compressing all images before testing.

We compare our method with the state-of-the-art NN-based Noiseprint [14], as well as demosaicing detection tools Bammey [3], Choi [12] (implementation of [4]) and Shin [54]. All methods, including ours, except Noiseprint operate on windows; we set their size to 32 × 32 pixels.

### 4.2. Results

Results are presented in Tab. 1 with the Matthews Correlation Coefficient (MCC), which is the cross-correlation coefficient between the ground truth and the detection. This metric, considered the most representative number for detection evaluation [10, 11], varies between -1 and 1, with 1 representing a perfect detection, -1 its complementary. Any random method has an expected score of 0. As all the tested methods produce heatmaps and not binary outputs, the test was done using the best threshold over each dataset.

On the Trace datasets, the method beats the state-of-the-art at all the tested compression levels. The results are similar to Bammey [3] on the uncompressed images, but our method presents a stronger robustness to JPEG compression. Choi [4, 12] presents some robustness to compression as well. Both Bammey and Shin [54] are unable to make any detection on compressed images. Noiseprint [14] is entirely blind to demosaicing pattern shifts, as seen in the Trace CFA Grid datasets. However, its positive results on the CFA Algorithm datasets mean that it is, to some extent, able to detect changes in the demosaicing algorithm used.

On the Korus dataset, our method presents the best results overall, despite images from the Canon 60D dataset presenting no demosaicing artefacts, as also evidenced by the other demosaicing detection methods. This may be due to an absence of any demosaicing or to a downsampling of the images – images are at a lower resolution than the camera’s maximal resolution. On the two Nikon cameras, we get MCC scores of 0.412 (Nikon D7000) and 0.408 (Nikon D90). The best score is reached on the Sony α57 camera, where the MCC is 0.628. All demosaicing detection methods perform best on that camera, which is probably due to a demosaicing that leaves more artefacts. Noiseprint is the only method able to provide relevant detections on the
Figure 7: Results on several images from the Korus [36, 37] (first 4 images) and Trace [5] CFA Grid (last 3 images) databases. From left to right: Forged image, forgery mask, detected demosaicing grids with the proposed methods, detected forgeries with the proposed method, and results of Noiseprint [14], Bammey [3], Shin [54] and Choi [4, 12]. As seen on the 1st and 2nd images, detecting the grid enables one not only to find forgeries, but also to precisely know how the forgery (here, an inpainting) was done, by localizing patches with different pattern alignments. On the 3rd image, no demosaicing traces are detected by any method. Still, the forgery can be detected by Noiseprint, which relies on different cues. On the 4th image, the same grid is detected on the whole image; the forged region cannot be detected by demosaicing analysis because the forged region’s pattern is aligned. On the last three rows, the same image has been processed uncompressed and at JPEG compression qualities 95 and 90. Even after compression, we can still detect the forgery.

<table>
<thead>
<tr>
<th>Method</th>
<th>Grid Exo</th>
<th>Grid Endo</th>
<th>Alg Exo</th>
<th>Alg Endo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>0.709</td>
<td>0.703</td>
<td>0.523</td>
<td>0.519</td>
</tr>
<tr>
<td>Bammey[3]</td>
<td>0.682</td>
<td>0.665</td>
<td>0.501</td>
<td>0.491</td>
</tr>
<tr>
<td>Shin[54]</td>
<td>0.104</td>
<td>0.099</td>
<td>0.085</td>
<td>0.084</td>
</tr>
<tr>
<td>Choi[4, 12]</td>
<td>0.603</td>
<td>0.575</td>
<td>0.420</td>
<td>0.385</td>
</tr>
<tr>
<td>Noiseprint[14]</td>
<td>-0.001</td>
<td>0.002</td>
<td>0.066</td>
<td>0.060</td>
</tr>
</tbody>
</table>

(a) Results on the CFA Grid (Grid) and CFA Algorithm (Alg) datasets of the Trace [5] database, with endomasks (Endo) and exomasks (Exo)

<table>
<thead>
<tr>
<th>Method</th>
<th>UNC</th>
<th>J95</th>
<th>J90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>0.709</td>
<td>0.307</td>
<td>0.151</td>
</tr>
<tr>
<td>Bammey[3]</td>
<td>0.682</td>
<td>0.005</td>
<td>0.003</td>
</tr>
<tr>
<td>Shin[54]</td>
<td>0.104</td>
<td>0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>Choi[4, 12]</td>
<td>0.603</td>
<td>0.156</td>
<td>0.070</td>
</tr>
<tr>
<td>Noiseprint[14]</td>
<td>-0.001</td>
<td>0.004</td>
<td>0.001</td>
</tr>
</tbody>
</table>

(c) Compared robustness to JPEG compression on Trace CFA Grid exo. UNC means no compression.

Table 1: Compared results on the Trace [5] and Korus [36, 37] datasets, using the MCC metric.
Canon 60D cameras. Although its final score is lower than the proposed method’s, we want to highlight that these two methods should be seen as complementary, not as competitors. As seen on the Trace database, Noiseprint is blind to shifts in the demosaicing pattern. Its detections on the Korus datasets are thus based on other kinds of artefacts. On the other hand, the proposed method focuses solely on demosaicing pattern shifts. Both thus provide a different and complementary insight into potential forgeries.

Visual results on different images can be seen on Fig. 7.

4.3. Ablation Study

We test the ability of the proposed method to detect forgeries when its key components are removed in Tab. 2. Although retraining is not strictly needed to analyse images, it improves the robustness of the network to various post-demosaicing processing (Korus) and JPEG compression (Trace). When the results are already clear without retraining, however, the difference is much more minor (uncompressed Trace). Unsurprisingly, pretraining the network on JPEG-compressed images is not necessary when analysing uncompressed images. By forcing the network to diversify its analysis, results are actually slightly lower on uncompressed datasets. Even on compressed images, the network that was not pretrained on JPEG images is still able to adapt to the compression to some extent. Nevertheless, the JPEG-pretrained network we propose is almost as good on uncompressed images, and much better when analysing compressed images. Unless one is certain the images were not compressed, the JPEG-pretrained network is thus better overall. Both JPEG pretraining and internal learning improve robustness to JPEG compression, and best results are achieved when both are used. Without at least one of them, however, the network is unable to make any detections on compressed images. In any case, when JPEG pretraining is performed, it must be done with the shifting strategy presented in Sec. 3.2. Otherwise, the network can be led to directly detect the JPEG pattern instead of detecting the mosaic over the JPEG compression, and its scores are worse than even the network without JPEG pretraining. The loss must also be computed on the four possible offset targets, and back-propagation must be performed on the best one globally. Without this, the network is forced to shift its detections into assuming the image is in the pattern, thus harming the results if this is not the case.

Discarding the diagonal pattern detection and only looking at the full patterns lowers the scores. Two thirds of the detectable inconsistencies feature different grids and the same diagonal, yet the diagonal is much easier to identify on more difficult (e.g. compressed) images or smaller inconsistencies, where a diagonal shift is often the only thing that can be detected significantly.

<table>
<thead>
<tr>
<th>Method</th>
<th>Korus</th>
<th>Trace</th>
<th>Trace J95</th>
<th>Trace J90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>0.353</td>
<td>0.709</td>
<td>0.307</td>
<td>0.151</td>
</tr>
<tr>
<td>No Retraining</td>
<td>0.267</td>
<td>0.625</td>
<td>0.143</td>
<td>0.036</td>
</tr>
<tr>
<td>Fixed Loss</td>
<td>0.178</td>
<td>0.472</td>
<td>0.099</td>
<td>0.028</td>
</tr>
<tr>
<td>No JPEG</td>
<td>0.362</td>
<td>0.728</td>
<td>0.254</td>
<td>0.083</td>
</tr>
<tr>
<td>No JPEG/Retraining</td>
<td>0.289</td>
<td>0.740</td>
<td>0.012</td>
<td>0.005</td>
</tr>
<tr>
<td>No Shifts</td>
<td>0.340</td>
<td>0.696</td>
<td>0.250</td>
<td>0.056</td>
</tr>
<tr>
<td>No Diagonal</td>
<td>0.189</td>
<td>0.281</td>
<td>0.125</td>
<td>0.060</td>
</tr>
</tbody>
</table>

Table 2: Ablation study of the method. **No JPEG**: pretraining was only done on uncompressed images. **No Retraining**: We directly use the pretrained network. **Fixed loss**: The target cannot be shifted to minimize the loss. **No Shifts**: the shifting JPEG described in Sec. 3.2 was not applied. **No Diagonal**: Detections were only made with the full pattern. **No Segmentation**: The score of each block is not maximized within each connected components.

5. Discussion

In this article, we have shown that positional and internal learning could be coupled to detect image forgeries as shifts in the image’s demosaicing pattern. The self-supervised nature of positional learning makes it fit for internal learning, as labels can be directly obtained from the tested image, considering it is authentic. The main difficulty that could be expected to come with single-image fine-tuning comes from overfitting, especially on forged regions. While an overfitting network will detect all pixels’ locations, and will thus fail to detect the forgery, it will not lead to a higher number of false positives. Overfitting is naturally limited by the small size of forgeries, combined with the fact that forgeries create evidence that contradict the rest of the image.

Our experiments show that our method detects demosaicing pattern shifts better than other demosaicing detection methods, and more generally beats the state of the art on the Korus dataset of uncompressed forged images.

Internal learning was performed with 15 iterations on each tested image. This number has been set heuristically, as results do not seem to improve much after that. In future work, obtaining an automatic stopping criterion would be desirable, to increase both the speed and detections.

We observed that demosaicing pattern detection is mostly undetected by more generic SOTA forensic algorithms, whereas our method specifically focuses on these. When trying to find forgeries, these methods yield complementary results and can thus work in parallel, not in competition. To this end, future work should focus on ways to interpret its results, to further limit its number of false detections and enable joint use with other methods.
References


[58] Shuiming Ye, Qibin Sun, and Ee-Chien Chang. Detecting digital image forgeries by measuring inconsistencies of

