IPOL IMAGE PROCESSING ON LINE

Published in Image Processing On Line on 2022–10–20.
Submitted on 2022–09–28, accepted on 2022–09–29.
ISSN 2105–1232 © 2022 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2022.431

Analysis and Experimentation on the ManTraNet Image Forgery Detector

Quentin Bammey

Université Paris-Saclay, ENS Paris-Saclay, Centre Borelli, Gif-sur-Yvette, France
quentin.bammey@ens-paris-saclay.fr

Communicated by Jean-Michel Morel *Demo edited by* Quentin Bammey

Abstract

This work describes the ManTraNet network for image forgery detection. ManTraNet is an end-to-end convolutional neural network composed of two sub-networks, one to extract features linked to traces of manipulation, and another to detect local anomalies between the features. It is trained on pristine and forged images from several datasets. We briefly analyze the results provided by ManTraNet, so as to highlight its qualities and limitations. Overall, ManTraNet yields state-of-the-art results on benchmark datasets with images similar to the one it sees in training, but is unreliable on wild images, due to its opacity and the difficulty distinguishing true detections from false positives.

Source Code

The source code and documentation for this algorithm are available from the web page of this article[1]. Usage instructions are included in the `README.txt` file of the archive. The original implementation of the method is available here[2].
This is an MLBriefs article, the source code has not been reviewed!

Keywords: image forensics; forgery detection; convolutional neural network

1 Introduction

Image forensics has become an important field of study over the past few years, sparked by the ubiquity of images on the internet and the proliferation of fake news in social media. Originally, image forgeries were mainly detected by manual methods targeting specific traces left by the image signal processing pipeline (ISP) such as demosaicing artifacts [3, 10, 13, 14, 20, 23, 27, 33, 35, 4], JPEG compression [1, 8, 18, 25, 26, 31, 32, 30], or noise inconsistencies [11, 15, 28, 29].

[1] https://doi.org/10.5201/ipol.2022.431
[2] https://github.com/RonyAbecidan/ManTraNet-pytorch

With the advent of deep learning, convolutional neural networks (CNN) were recently introduced to image forensics. They can be trained on pristine images so as to detect whether two patches could come from the same image or might have been processed differently, indicating a forgery [12, 17]. Some networks are introduced to improve the analysis of a specific point, such as demosaicing artifacts [2, 6]. ManTraNet [39], which we study here, pioneers a third category of methods, that are trained directly on forged images to localize their forged regions.

The ability of methods such as ManTraNet to provide exceptional results on benchmark datasets has already been demonstrated. However, in such datasets, neural networks can be trained on images and forgeries similar to those of the evaluation set. One can wonder whether such performances hold in the wild, where images are diverse and largely differ from the controlled environment of training datasets. Furthermore, the interpretability of the results can be questioned; as the reason behind the detection is not immediately clear, such results can often be seen as opaque.

After a brief description of ManTraNet, we will analyze its results on various images to question the performances in uncontrolled scenarios. While highlighting the interpretability issue of ManTraNet, we will also conduct short experiments to try to understand what triggers detections.

We use the pytorch re-implementation from https://github.com/RonyAbecidan/ManTraNet-pytorch, which is equivalent and uses the same weights and network structure as the original authors' [39] repository, available at https://github.com/ISICV/ManTraNet. Note that even in the official implementation, the network architecture is slightly different than described in the paper, this discrepancy has already been acknowledged by the authors. The structure described here corresponds to the one that was actually implemented.

2 ManTraNet

In this section, we briefly detail the ManTraNet method.

2.1 Network Architecture

ManTraNet is an end-to-end CNN, whose architecture is presented in Figure 1. The input is a color image. The output is a one-channel heatmap with floating scores between 0 and 1 (or integer scores between 0 and 255 after saving the result as an image), representing the confidence in each position belonging to a forged region. Scores closer to 1 (or 255) correspond to areas which ManTraNet is very confident have been forged, whereas scores closer to 0 correspond to regions considered pristine.

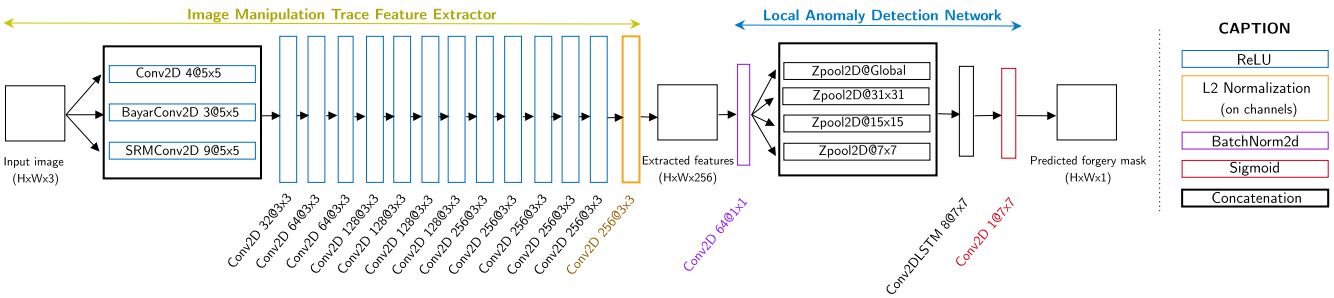


Figure 1: Overview of the ManTraNet architecture. Figure from https://github.com/RonyAbecidan/ManTraNet-pytorch.

The network itself is composed of 2 sub-networks. The first sub-network aims to extract manipulation traces from the image and follows a VGG [36] architecture. It takes the image as input and

outputs 256 feature maps of the same size, which represent features related to different types of manipulations. A first layer consists of 16 5×5 convolutions to extract initial features from the image. 4 of those convolutions are classic convolutions, 9 are style-based recalibration modules (SRM) [24] and the remaining three are Bayar constrained convolutions[3] [7]. The 16 features are concatenated and followed by 3×3 convolutions layers; in order, one layer with 32 feature channels, 2 with 64 channels, 4 with 128 channels and 6 with 256 channels. The output of the last layer is used as the extracted features. All convolutions are followed by ReLU activations, except for the last layer which is processed by L_2 normalization.

The second sub-network takes the feature map as input, and outputs a single-channel map representing the confidence in each position being forged. The 256 feature maps are first summarized and adapted into 64 maps with a pointwise convolution layer followed by a batch normalization. For each feature map, ManTraNet computes the deviation of each pixel from the dominant feature of this map. If F is the feature map, the globally dominant feature μ_{-1} is the average of all values of F

$$\mu_{-1}[x, y] \triangleq \mu_{-1} \triangleq \frac{1}{XY} \sum_{i=0}^{X-1} \sum_{j=0}^{Y-1} F[i, j], \tag{1}$$

where (X, Y) is the size of the image. The locally dominant feature at resolution n (for non-negative odd n) μ_X^n is computed as the local average within a $n \times n$ window centered at each pixel

$$\mu_n[x, y] \triangleq \frac{1}{n^2} \sum_{i=\frac{n-1}{2}}^{\frac{n+1}{2}} \sum_{j=\frac{n-1}{2}}^{\frac{n+1}{2}} F[x + i, y + j]. \tag{2}$$

The normalized deviation of each pixel from one of these dominant features is then

$$Z_n[x, y] = \frac{F[i, j] - \mu_n[x, y]}{\max\left(\sigma_F, \epsilon + \omega\right)}, \tag{3}$$

for non-negative odd n (local scale) or $n = -1$ (global resolution), where σ_F is the standard deviation of F, $\epsilon = 10^{-5}$ and ω is a non-negative weight learned independently for each feature map.

This process, known as Z-pooling, is performed at the global scale ($n = -1$) and at resolutions $n \in [7, 15, 31]$. The rationale for using local scales in addition to the global average is to mitigate the influence of multiple forgeries in an image. The four resolutions are concatenated along an artificial time axis, and the $4 \times 64 \times X \times Y$ tensor is passed through a convolutional LSTM [34], that looks at the features starting from the global resolution and moves towards more local resolutions if it is not certain at the current level. The convolutional LSTM uses 7×7 convolutions and returns 8 feature maps. A final 7×7 convolution followed by a sigmoid activation yields the single-channel heatmap.

2.2 Training

Here we describe the training of the network in the original paper. We focus on analyzing the results of the method and do not recreate the training ourselves, as details for training are not provided in the paper nor in the implementation.

The first sub-network is trained using the Dresden Image Database [16] of pristine images. Of all the images, 80% are used for training, 10% for validation and the remaining 10% for testing. Images are divided into 256×256 patches, and homogeneous patches with intensity standard deviation below $\frac{32}{255}$ are rejected. In total, 1.25M patches are kept. Samples are generated by uniformly sampling a

[3] This description corresponds to the actual architecture of the official implementation. The original paper itself [39] instead describes 10 classic convolutions, 3 SRM and three Bayar-constrained convolutions.

random patch and manipulation, applying the manipulation to the image, and cropping a random 128×128 region of the output. The network is fed with the cropped manipulated region, and trained to identify which manipulation was used. Details on the module for auxiliary training are not provided in the original paper.

The second sub-network is trained on four synthetic datasets: [37], [38], a dataset synthesized by using OpenCV [9] inpainting on the Dresden images [16], and another dataset obtained by locally applying random manipulations on the same images. Training is done on patches of size 256×256.

Both networks are trained with a batch size of 64, 1000 batches per epoch, using the Adam [19] optimizer with an initial learning rate of 10^{-4} without decay. The learning rate is halved if the validation loss fails to improve for 20 epochs. The epoch with the best validation loss is kept for the final model.

3 Experiments

In this section, we conduct several experiments: we test ManTraNet on authentic images to test its robustness to false positives, as well as on forgeries, both real and from controlled environments, to see whether it responds to the forgeries. Finally, we investigate to which changes the method is sensitive.

3.1 Response to Authentic Images

We first test the detections of the method on authentic images. The Korus database [21, 22] contains 220 tampered images of size 1920×1080 from 4 different cameras, and their associated pristine images. We run ManTraNet on the 220 pristine images, and check the proportion of images that contain false positive connected regions over a given size where the output is above a given confidence threshold. Results can be seen in Table 1. These observations on pristine images show the difficulty of identifying forgeries with ManTraNet; even at a threshold of 0.9, more than half of the pristine images contain a detection over 256 pixels (corresponding to a 16×16 square region). In other words, even if ManTraNet detects a small region with very high confidence, it is impossible to consider this detection reliable, as many (authentic) regions will be detected with the same confidence. Larger forgeries might more easily be identified, nonetheless there are still large pristine regions that are detected with high confidence: detections from the method cannot be considered fully reliable. Furthermore, as seen in visual results in Figure 2, even though the noise-like false positive responses can be rejected by a trained eye, the strongly-confident responses on some regions cannot be reliably distinguished from real detections.

3.2 Detection of Forgeries

In Figure 3, we show the response of ManTraNet to forged images from the Korus Dataset [21, 22]. First of all, we note that the ManTraNet strongly responds to many forgeries, which can help in their detection. That being said, the response of ManTraNet to those forgeries is not unlike its response to pristine images as those from Figure 2. Given only the output from ManTraNet, it is thus difficult to distinguish true forgeries from false positives, even to a trained eye.

To complicate the matter, the tested images correspond to controlled forgeries from a dataset. In real cases, images often undergo much more post-processing and editing, which can make their detection different or more difficult than on benchmark datasets, where forgeries are closer in nature to those on which the method has been trained. In Figure 4, we can indeed see that ManTraNet is unable to detect most of the tested real cases of forgeries.

Threshold	0.5	0.6	0.7	0.8	0.9	0.95
64 pixels	100%	100%	99%	98%	91%	74%
128 pixels	100%	99%	97%	91%	78%	58%
256 pixels	97%	93%	89%	82%	59%	40%
512 pixels	90%	85%	78%	60%	42%	26%
1024 pixels	79%	68%	54%	44%	28%	18%
2048 pixels	59%	48%	40%	31%	16%	12%
4096 pixels	45%	35%	28%	20%	12%	8%
8192 pixels	30%	25%	20%	12%	8%	2%
16384 pixels	20%	15%	11%	8%	4%	2%

Table 1: Percentage of pristine images of the Korus dataset for which ManTraNet outputs a confidence over a given threshold within a connected region over a given size, in pixels. Images are 1920×1080 of size.

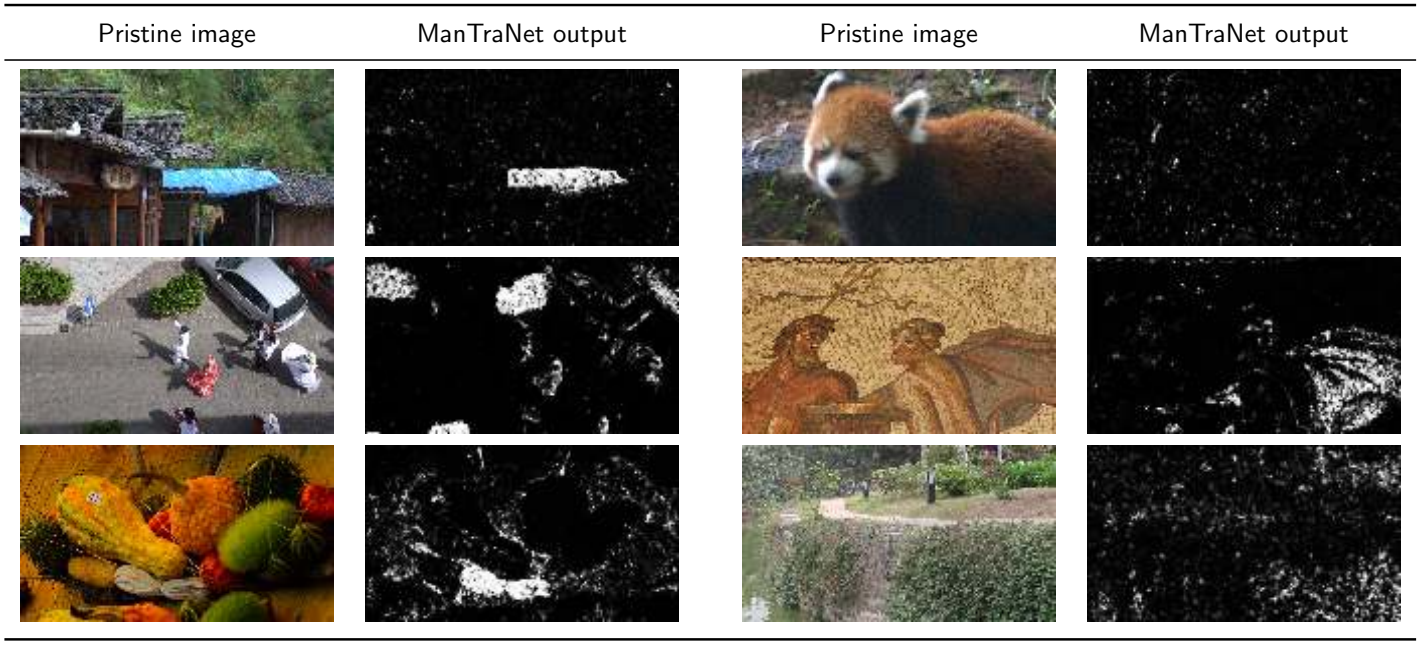


Figure 2: Example detections of ManTraNet on pristine images from the Korus [21, 22] dataset. Although the noise-like false positive responses can be rejected by a trained eye, the strongly-confident output on some regions cannot be reliably distinguished from real detections.

3.3 Interpretability

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: the region inside the forgery mask can have a different demosaicing pattern (`cfa_grid` dataset), a different demosaicing altogether (`cfa_alg`), a different JPEG grid (`jpeg_grid` dataset), a completely different JPEG compression (`jpeg_quality`), different noise levels on the raw image (`noise`) or a combination of those changes (`hybrid`). The forgery masks are selected randomly from a semantic segmentation of the image[4]. Because the only change between the authentic and forged images is the processing inside a mask, detections cannot be made for purely semantic reasons. The

[4]The database also exists with exogenous forgery masks taken from other images; both are equivalent for our use case so we only consider the endogenous masks.

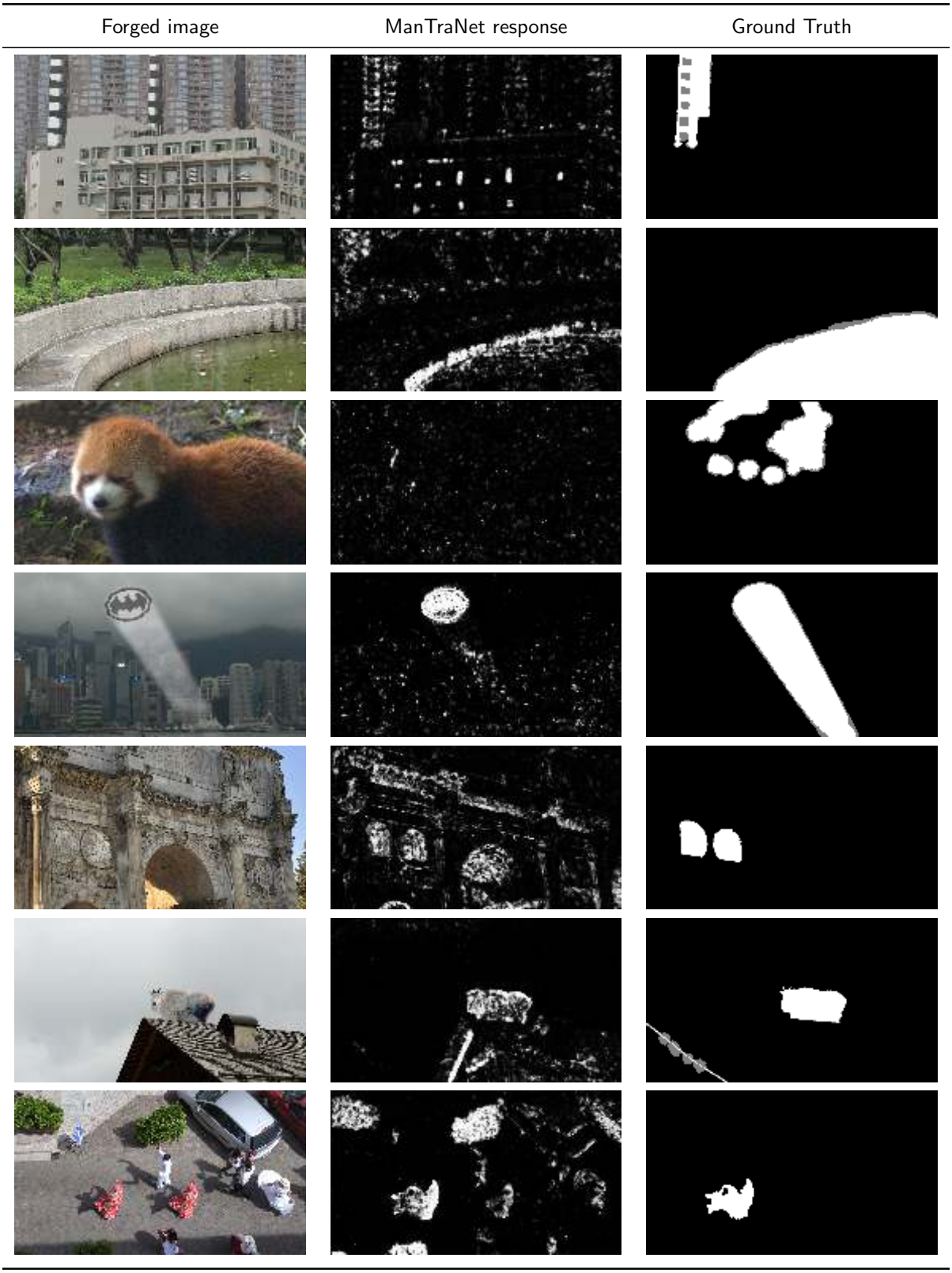


Figure 3: ManTraNet response to forged images from the Korus Dataset [21, 22]. In many cases, ManTraNet is sensitive to the forgeries. However, it also responds to other, authentic regions in the image. In the last image, the forgery is an internal copy-move. Note that ManTraNet also responds to the original (authentic) region, albeit not as strongly. Overall, even the true positives are difficult to distinguish from false positives from Figure 2.

structure of this database enables us to see whether ManTraNet is sensitive to the different changes in processing. We report the scores in Table 2. As can be seen, ManTraNet is entirely insensitive to

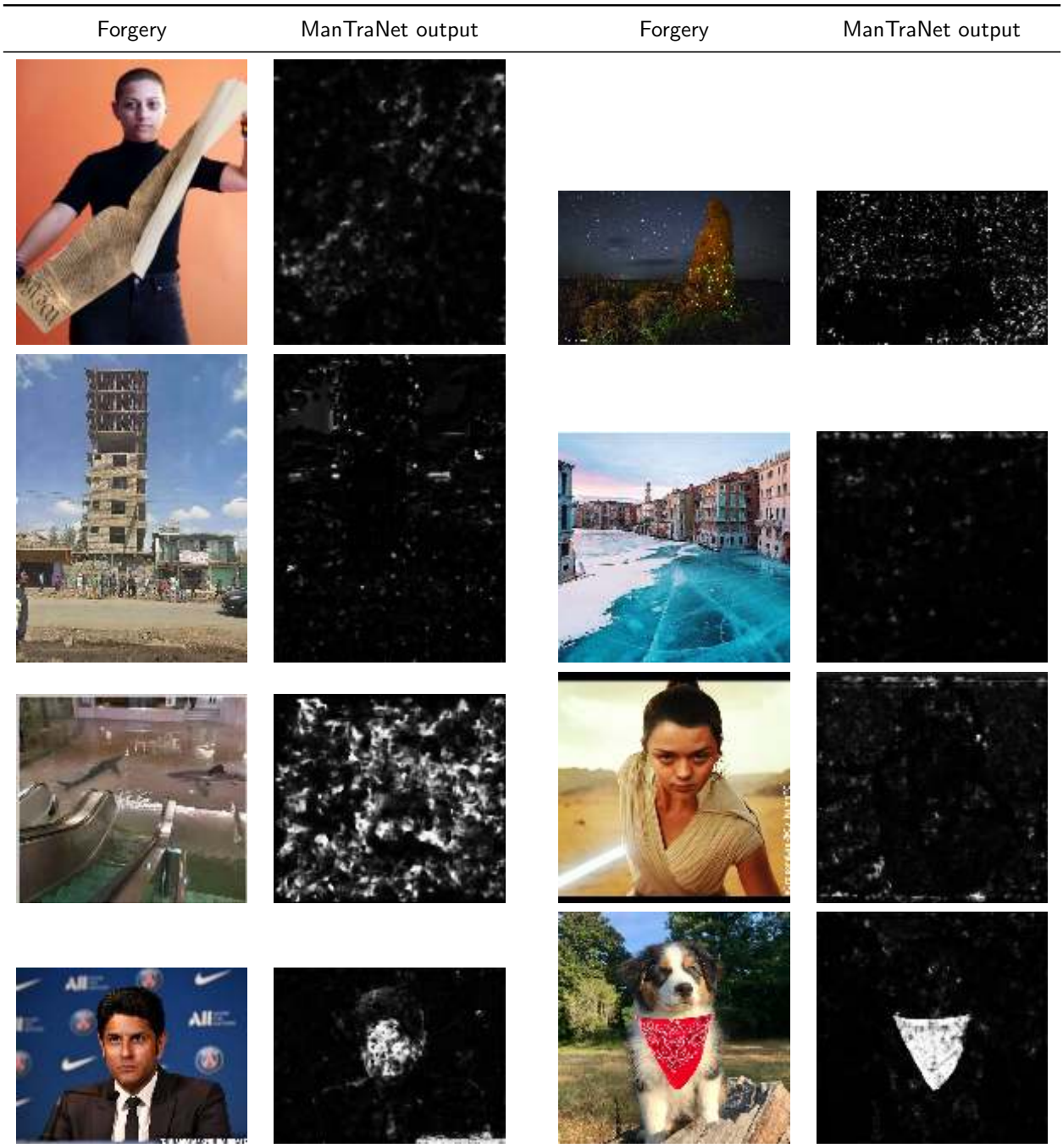


Figure 4: Response to several real-case forgeries. ManTraNet is able to detect the dog's added bandana and one of the @GuillaumeTC images, but is unable to detect the other forgeries, that can differ a lot from forgeries seen in training environments. See the last section for image credits.

shifts in the traces of the image, such as shifts in the mosaic (CFA Grid) or in the JPEG grid. It is somehow sensitive to inconsistencies of the noise level, demosaicing algorithm and JPEG compression quality, but this sensitivity remains low compared to the standard deviation of the scores.

Overall, the grounds for detection by ManTraNet remain obscure, as only a small part of the reasons can be explained by the analysis with the Trace dataset. We note, though, that this lack of interpretability is not fatal. Indeed, the first part of the network is trained to identify manipulations among a large palette of potential changes. The output of the auxiliary network, if it were made public, might enable a much better interpretability of the results, as it would be possible to know at least why the network responds positively to some regions.

Dataset	MCC	MCC standard deviation
Noise	0.032	0.099
CFA Grid	-0.004	0.065
CFA Alg	0.053	0.165
JPEG Grid	0.000	0.043
JPEG Quality	0.086	0.171
Hybrid	0.107	0.176

Table 2: For each dataset of the Trace [5] database, we report the Matthew's Correlation Coefficient (MCC) of ManTraNet, taken with the globally-best threshold for each dataset. The MCC takes values between -1 and 1, with 1 representing a perfect detection, -1 its complementary. As a baseline, a random classifier or input-independent method is expected to yield a score of 0. The globally-best threshold is selected over each dataset, then the MCC is computed separately for each image and averaged. We also provide the standard deviation of the score over the dataset. As can be seen, ManTraNet is entirely insensitive to shifts in the traces of the image, such as shifts in the mosaic (CFA Grid) or in the JPEG grid. It is somehow sensitive to inconsistencies of the noise level, demosaicing algorithm and JPEG compression quality, but this sensitivity remains low compared to the standard deviation of the scores.

4 Conclusion

In this work, we briefly described the ManTraNet forgery detection network, and analyzed its results in different cases. While ManTraNet is able to respond to many forgeries in datasets, it fails to do so in controlled environments. Furthermore, it is highly sensitive to even pristine regions, and lacks a method for automatic detection – the network only outputs a confidence heatmap. This creates difficulties in distinguishing true forgeries from false positives in the output of the method, even to expert eyes. Finally, the reasons for which a strong response is output by the end-to-end network remain obscure. Access to the auxiliary network in the middle of the method, used in training to different kinds of forgeries, could provide insight into the reasons for detection, and ultimately more interpretability at least to experts. Overall, the lack of interpretability and robustness to false positive seem to limit the role ManTraNet can play in image forensics. Nevertheless, it remains a useful tool to help localize or confirm forgeries detected by other means, as well as to hightlight suspicious regions for analysis by other methods.

Acknowledgements

This work has received funding by the European Union under the Horizon Europe vera.ai project, Grant Agreement number 101070093, and also as part of the ANR/DGA DEFALS challenge, grant ANR-16-DEFA-0004 Signature d'Images.

Image Credits

 Photo by Tina Nikoukhah.

 Biosphoto/Alamy Stock Photo.

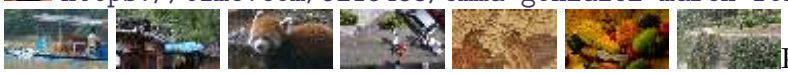
 Photo by Jamie King (c) 2012.

  Photo by GuillaumeTC https://twitter.com/GuillaumeTC.

 Forgery reported by https://factcheck.afp.com/no-not-photo-dangerous-building-under-construction

 Photo by Robert Jahn.

 https://time.com/5215433/emma-gonzalez-march-for-our-lives-fake-photo/.

 Korus dataset [21, 22], pristine images https://pkorus.pl/downloads/dataset-realistic-tampering

 Korus dataset [21, 22], forged images https://pkorus.pl/downloads/dataset-realistic-tampering

References

[1] I. Amerini, R. Becarelli, R. Caldelli, and A. Del Mastio, *Splicing forgeries localization through the use of first digit features*, in IEEE International Workshop on Information Forensics and Security (WIFS), IEEE, 2014, pp. 143–148. https://doi.org/10.1109/WIFS.2014.7084318.

[2] Q. Bammey, R. Grompone von Gioi, and J-M. Morel, *An adaptive neural network for unsupervised mosaic consistency analysis in image forensics*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 6 2020. https://doi.org/10.1109/CVPR42600.2020.01420.

[3] ——, *Image Forgeries Detection through Mosaic Analysis: the Intermediate Values Algorithm*, Image Processing On Line, 11 (2021), pp. 317–343. https://doi.org/10.5201/ipol.2021.355.

[4] ——, *Demosaicing to detect demosaicing and image forgeries*, (2022).

[5] Q. Bammey, T. Nikoukhah, M. Gardella, R. Grompone von Gioi, M. Colom, and J-M. Morel, *Non-semantic evaluation of image forensics tools: Methodology and database*, in IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 1 2022, pp. 3751–3760. https://doi.org/10.1109/WACV51458.2022.00244.

[6] R. Bammey, Q.and Grompone von Gioi and J-M. Morel, *Forgery detection by internal positional learning of demosaicing traces*, in IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 1 2022, pp. 328–338. https://doi.org/10.1109/WACV51458.2022.00109.

[7] B. Bayar and M.C. Stamm, *Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection*, IEEE Transactions on Information Forensics and Security, 13 (2018), pp. 2691–2706. https://doi.org/10.1109/TIFS.2018.2825953.

[8] T. Bianchi, A. De Rosa, and A. Piva, *Improved DCT coefficient analysis for forgery localization in JPEG images*, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2011, pp. 2444–2447. https://doi.org/10.1109/ICASSP.2011.5946978.

[9] G. Bradski, *The OpenCV Library*, Dr. Dobb's Journal of Software Tools, (2000).

[10] C-H. Choi, J-H. Choi, and H-K. Lee, *CFA pattern identification of digital cameras using intermediate value counting*, in ACM Multimedia Workshop on Multimedia and Security, New York, NY, USA, 2011, ACM, pp. 21–26. https://doi.org/10.1145/2037252.2037258.

[11] D. Cozzolino, G. Poggi, and L. Verdoliva, *Splicebuster: A new blind image splicing detector*, in IEEE International Workshop on Information Forensics and Security (WIFS), IEEE, 2015, pp. 1–6. https://doi.org/10.1109/WIFS.2015.7368565.

[12] D. Cozzolino and L. Verdoliva, *Noiseprint: A CNN-Based Camera Model Fingerprint*, IEEE Transactions on Information Forensics and Security, 15 (2020), pp. 144–159. https://doi.org/10.1109/TIFS.2019.2916364.

[13] A.E. Dirik and N. Memon, *Image tamper detection based on demosaicing artifacts*, in IEEE International Conference on Image rocessing (ICIP), IEEE, 2009, pp. 1497–1500. https://doi.org/10.1109/ICIP.2009.5414611.

[14] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, *Image forgery localization via fine-grained analysis of CFA artifacts*, IEEE Transactions on Information Forensics and Security, 7 (2012), pp. 1566–1577. https://doi.org/10.1109/TIFS.2012.2202227.

[15] M. Gardella, P. Musé, J-M. Morel, and M. Colom, *Noisesniffer: a fully automatic image forgery detector based on noise analysis*, in IEEE International Workshop on Biometrics and Forensics (IWBF), IEEE, 2021, pp. 1–6. https://doi.org/10.1109/IWBF50991.2021.9465095.

[16] T. Gloe and R. Böhme, *The 'Dresden Image Database' for benchmarking digital image forensics*, in ACM Symposium On Applied Computing, vol. 2, 2010, pp. 1585–1591. https://doi.org/10.1145/1774088.1774427.

[17] M. Huh, A. Liu, A. Owens, and A.A. Efros, *Fighting fake news: Image splice detection via learned self-consistency*, in European Conference on Computer Vision (ECCV), 9 2018, pp. 101–117. https://doi.org/10.1007/978-3-030-01252-6_7.

[18] C. Iakovidou, M. Zampoglou, S. Papadopoulos, and Y. Kompatsiaris, *Content-aware detection of JPEG grid inconsistencies for intuitive image forensics*, Journal of Visual Communication and Image Representation, 54 (2018), pp. 155–170. https://doi.org/10.1016/j.jvcir.2018.05.011.

[19] D.P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2014. https://doi.org/10.48550/arXiv.1412.6980.

[20] M. Kirchner and J. Fridrich, *On detection of median filtering in digital images*, in Media forensics and security II, vol. 7541, International Society for Optics and Photonics, 2010, p. 754110. https://doi.org/10.1117/12.839100.

[21] P. Korus and J. Huang, *Evaluation of random field models in multi-modal unsupervised tampering localization*, in IEEE International Workshop on Information Forensics and Security (WIFS), 2016. https://doi.org/10.1109/WIFS.2016.7823898.

[22] ——, *Multi-scale Analysis Strategies in PRNU-based Tampering Localization*, IEEE Transactions on Information Forensics and Security, (2017). https://doi.org/10.1109/TIFS.2016.2636089.

[23] N. Le and F. Retraint, *An improved algorithm for digital image authentication and forgery localization using demosaicing artifacts*, IEEE Access, 7 (2019), pp. 125038–125053. https://doi.org/10.1109/ACCESS.2019.2938467.

[24] H. Lee, H-E. Kim, and H. Nam, *SRM: A Style-Based Recalibration Module for Convolutional Neural Networks*, in IEEE/CVF International Conference on Computer Vision (ICCV), 10 2019. http://dx.doi.org/10.1109/ICCV.2019.00194.

[25] W. Li, Y. Yuan, and N. Yu, *Passive detection of doctored JPEG image via block artifact grid extraction*, Signal Processing, 89 (2009), pp. 1821–1829. https://doi.org/10.1016/j.sigpro.2009.03.025.

[26] Z. Lin, J. He, X. Tang, and C-K. Tang, *Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis*, Pattern Recognition, 42 (2009), pp. 2492–2501. https://doi.org/10.1016/j.patcog.2009.03.019.

[27] L. Liu, Y. Zhao, R. Ni, and Q. Tian, *Copy-Move Forgery Localization Using Convolutional Neural Networks and CFA Features*, International Journal of Digital Crime and Forensics (IJDCF), 10 (2018), pp. 140–155. https://doi.org/10.4018/IJDCF.2018100110.

[28] S. Lyu, X. Pan, and X. Zhang, *Exposing region splicing forgeries with blind local noise estimation*, International Journal of Computer Vision, 110 (2013), pp. 202–221. https://doi.org/10.1007/s11263-013-0688-y.

[29] B. Mahdian and S. Saic, *Using noise inconsistencies for blind image forensics*, Image and Vision Computing, 27 (2009), pp. 1497–1503. https://doi.org/10.1016/j.imavis.2009.02.001.

[30] T. Nikoukhah, J. Anger, M. Colom, J-M. Morel, and R. Grompone von Gioi, *ZERO: a Local JPEG Grid Origin Detector Based on the Number of DCT Zeros and its Applications in Image Forensics*, Image Processing On Line, 11 (2021), pp. 396–433. https://doi.org/10.5201/ipol.2021.390.

[31] T. Nikoukhah, J. Anger, T. Ehret, M. Colom, J-M. Morel, and R. Grompone von Gioi, *JPEG grid detection based on the number of DCT zeros and its application to automatic and localized forgery detection*, in IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 110–118. https://openaccess.thecvf.com/content_CVPRW_2019/papers/Media%20Forensics/Nikoukhah_JPEG_Grid_Detection_based_on_the_Number_of_DCT_Zeros_CVPRW_2019_paper.pdf.

[32] T. Nikoukhah, M. Colom, J-M. Morel, and R. Grompone von Gioi, *Local JPEG Grid Detector via Blocking Artifacts, a Forgery Detection Tool*, Image Processing On Line, 10 (2020), pp. 24–42. https://doi.org/10.5201/ipol.2020.283.

[33] A. C. Popescu and H. Farid, *Exposing digital forgeries in color filter array interpolated images*, IEEE Transactions on Signal Processing, 53 (2005), pp. 3948–3959. https://doi.org/10.1109/TSP.2005.855406.

[34] X. Shi, Z. Chen, H. Wang, D-Y. Yeung, W-K. Wong, and W-C. Woo, *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*, in Advances in Neural Information Processing Systems, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds., vol. 28, Curran Associates, Inc., 2015. https://proceedings.neurips.cc/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

[35] H.J. Shin, J.Ju. Jeon, and I.K. Eom, *Color filter array pattern identification using variance of color difference image*, Journal of Electronic Imaging, 26 (2017), pp. 1 – 12. https://doi.org/10.1117/1.JEI.26.4.043015.

[36] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014. https://doi.org/10.48550/arXiv.1409.1556.

[37] Y. Wu, W. Abd-Almageed, and P. Natarajan, *Deep matching and validation network: An end-to-end solution to constrained image splicing localization and detection*, in ACM international Conference on Multimedia, 2017, pp. 1480–1502. https://doi.org/10.1145/3123266.3123411.

[38] ——, *BusterNet: Detecting copy-move image forgery with source/target localization*, in European Conference on Computer Vision (ECCV), 2018, pp. 168–184. https://doi.org/10.1007/978-3-030-01231-1_11.

[39] ——, *ManTra-Net: Manipulation Tracing Network for Detection and Localization of Image Forgeries With Anomalous Features*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6 2019. https://doi.org/10.1109/CVPR.2019.00977.