AFRL-RI-RS-TR-2021-055

DIGITAL, SEMANTIC AND PHYSICAL ANALYSIS OF MEDIA INTEGRITY

UNIVERSITY OF SOUTHERN CALIFORNIA, INFORMATION SCIENCES INSTITUTE (USC-ISI)

MARCH 2021

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2021-055 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
JEFFREY CARLO
Work Unit Manager

/ S /
JAMES PERRETTA
Deputy Chief, Information
Exploitation & Operations Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MARCH 2021	FINAL TECHNICAL REPORT	MAY 2016 - SEP 2020

4. TITLE AND SUBTITLE

DIGITAL, SEMANTIC AND PHYSICAL ANALYSIS OF MEDIA INTEGRITY

5a. CONTRACT NUMBER
FA8750-16-2-0204

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62303E

6. AUTHOR(S)

Wael Abd-Almageed (USC/ISI)
Ram Nevatia, (USC/ISI)
Annalisa Verdoliva (UNINA)
Christian Riess (FAU)

5d. PROJECT NUMBER
MEDI

5e. TASK NUMBER
40

5f. WORK UNIT NUMBER
08

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Prime: USC/ISI - 4676 Admiralty Way, MdR, CA 90292

Sub: Università degli Studi di Napoli Federico II (UNINA) - Via Claudio, 21, Napoli, Italy 80125
Sub: Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) - Martensstr.3, Erlangen, Germany 91058

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIGC
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2021-055

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this report, we present summary of the digital, physical and semantic image forensics and integrity methods developed by the DiSPARITY team, lead by the University of Southern California Information Sciences Institute, under DARPA's Media Forensics program between 2016 and 2020. The team also included University of Naples, Italy and University of Erlangen-Nuremberg, Germany. The DiSparity team has developed various state of the digital integrity methods (e.g. NoisePrint, GAN fingerprint and ManTra-Net), physical integrity (e.g. segmentation-free light direction estimation and analysis of incident light direction).

15. SUBJECT TERMS

Digital integrity, physical integrity, semantic integrity, deepfake detection, image manipulation detection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**JEFFREY CARLO**
U	U	U	UU	48	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

TABLE OF CONTENTS

List of Figures

List of Tables

1 Abstract

In this report, we present summary of the digital, physical and semantic image forensics and integrity methods developed by the DiSPARITY team, lead by the University of Southern California Information Sciences Institute, under DARPA's Media Forensics program between 2016 and 2020. The team also included University of Naples, Italy and University of Erlangen-Nuremberg, Germany. The DiSparity team has developed various state of the digital integrity methods (e.g. NoisePrint, GAN fingerprint and ManTra-Net), physical integrity (e.g. segmentation-free light direction estimation and analysis of incident light direction).

2 Summary

In this report, we present the novel methods we proposed and implemented for digital integrity, physical integrity and semantic integrity. For the research of digital integrity, we developed algorithms for camera fingerprint modeling and manipulation detection, including:

- A deep learning-based noiseprint to represent the camera characteristics, and trained Siamese network and classifier to perform camera identity matching and classification respectively with state of the art results.
- A GAN to generate the noiseprints of cameras.
- A variety of algorithms and models to perform image-based manipulation detection, copy/move localization, splice localization and video facial manipulation detection resulting in top-rated performance among Medifor evaluation participants. A notably algorithm is the ManTraNet: an end-to-end trainable image manipulation detection algorithm.
- A two-branch Recurrent Network for Isolating Deepfakes in Videos. Evaluation results showed better performance than all existing methods.

Our research of physical integrity focused on two directions — (1) incidental light direction estimation and (2) novel approaches to fingerprinting the camera and imaging process. We developed physical integrity algorithms, including:

- Two light direction estimation methods (gradient-based and convolutional network-based) are designed and evaluation results showed very good splice detecting performance when light direction estimation is applied.
- A number of fingerprints for describing camera characteristics, such as color fingerprinting (color filter sensitivity, camera white-balancing and additional camera-internal non-linearities like gamma correction.), fingerprinting of JPEG library chroma subsampling, and fingerprinting of depth image calculation in Cameras.

For semantic integrity, we have designed an end-to-end system for indexing a large image database, and retrieving provenance images of a given probe image. The evaluation results showed our system ranked the second in terms of provenance filtering performance among three participants. For a million-image world set, our system can reliably retrieve over 80% provenance images when looking at top-ranked candidates. This showed the system is promising for a real-world application.

3 Introduction

For digital and physical image integrity, we focused on modeling and finding manipulations in images or videos. For physical integrity, we focused on searching an image collection for original images and intermediate images generated when making a manipulated image using image editing tools.

Digital integrity tends to care about clues of the manipulation that exist in the image itself, e.g., whether the PRNU pixels do not match those of the claimed camera, or whether a trained neural network can indicate the manipulation as an abnormal region from a heat-map. Although the PRNU of a camera is stable, when capturing a natural scene image, it is always made obscure due to the relatively low intensity. Thus, when comparing the camera fingerprints between two natural scene images, it is beneficial to apply metric learning or complex classifiers than simply computing the distance between the fingerprints. Some types of manipulations can be detected as pixel-level local or global modifications. However, manipulations like the copy/move and splicing need to be detected by examining if duplicated regions exist in the image. As DeepFake becomes a powerful tool to fake images, there has been an imperative need of algorithms detecting DeepFake-generated manipulations. The key to developing these algorithm is to use deep learning to extract coherent features that can capture distinguishable distributions between genuine images and manipulated images.

Same as digital integrity, research in physical integrity also focuses on finding manipulations from images or videos. But the subject to research on is very different from that of digital integrity. For example, when splicing an object taken from one image into another image, it is very difficult to modify the direction of the light source lighting the object. This fact can be useful for splice detection. As one can see, the direction of the light can be considered an attribute intrinsic to a physical process. Intrinsic

processes of a camera such as how the signal received in the sensor is converted into a digital image, how Gamma compression parameters and image encoding are selected are also characteristics of the camera. All of those characteristics can be useful in media forensic.

The goal of provenance filtering is finding evidence of manipulation by means of image retrieval. Given a probe image, if one can find the original images, and even the intermediate images created during manipulation, the manipulation will look very obvious when one compares all these "provenance" images against each other. One challenge in building a provenance filtering algorithm is that the boundary of the retrieval query is not precisely given (it can be just an arbitrary and quite small region within the probe image). What's more, one cannot run an ordinary object detection network to solve this problem because the training data are not sufficient and the time to perform the retrieval will be too slow when feeding all images from the world dataset into the network.

4 Methods, Assumptions and Procedures

4.1 Digital Integrity

4.1.1 Noiseprint

Forensic analyses of digital images rely heavily on the traces of in-camera and out-camera processes left on the acquired images. Such traces represent a sort of camera fingerprint. If one is able to recover them, by suppressing the high-level scene content and other disturbances, a number of forensic tasks can be easily accomplished. A notable example is the Photo-Response Non-Uniformity (PRNU) pattern, which can be regarded as a device fingerprint, and has received great attention in multimedia forensics. Following this line of reasoning, we proposes a method to extract a camera *model* fingerprint, called noiseprint, where the scene content is largely suppressed and model-related artifacts are enhanced [15].

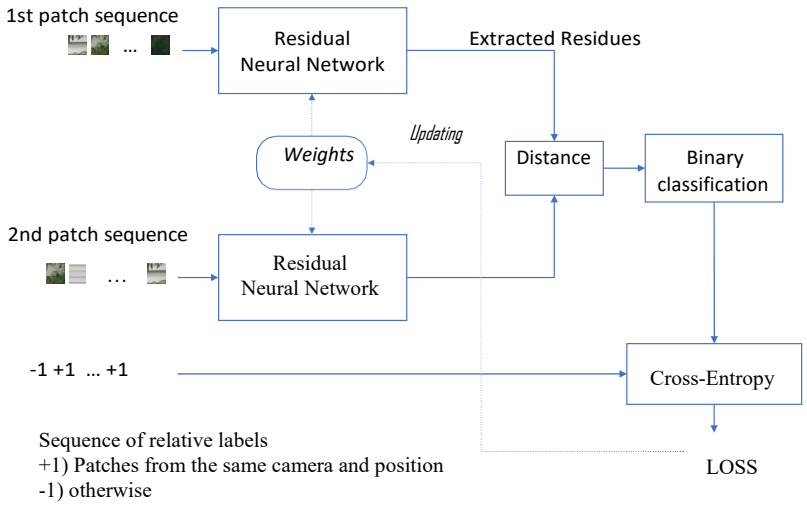


Figure 1: Using a Siamese architecture for training. The output of one CNN takes the role of desired (same model and position) or undesired (different models or positions) reference for the other twin CNN.

This is obtained by means of a Siamese network, shown above, which is trained with pairs of image patches coming from the same camera (label +1) or different cameras (label _1). By so doing, we obviate the absence of the noiseprint, which needs be estimated. In fact, if two different input patches acquired with the same camera model are fed to the two branches, their outputs are expected to be similar, and hence the output of net 1 can take the role of desired output for the input of net 2, and vice-versa, providing two reasonable input-output pairs. For both nets, we can therefore compute the error between the real output and the desired output, and back-propagate it to update the network weights. More in general, for positive examples (same model) weights are updated so as to reduce the distance between the outputs, while for negative examples (different models) weights are updated to increase this distance so the network learns to discard irrelevant information, common to all models, and keep in the noiseprint only the most discriminative features.

A further key point in the training process is that two input patches can be considered similar only if they come from the same position in the image, besides coming from the same camera model. In fact, artifacts generated by in-camera processes are not spatially stationary, and hence noiseprint patches

corresponding to different positions are different themselves and must not be pooled during training, in order not to dilute the artifacts' strength. An important consequence for forensic analyses is that any image shift, not to talk of rotation, will impact on the corresponding noiseprint, thereby allowing for the detection of many types of manipulations.

The loss function used for training is the weighted sum of two terms

$$\mathsf{L} = \mathsf{L}_0 - \lambda \mathsf{R} \tag{1}$$

with the weight λ to be determined by experiments. As first term we use the distance based logistic, L_0, recently proposed in the literature for similar tasks. Then a regularization term, R, based on average the spectral content of extracted noiseprints, is added to encourage their diversity.

4.1.2 An end-to-end trainable approach for image forgery detection

Due to limited computational and memory resources, current deep learning models accept only rather small patches in input, much smaller than typical images. In computer vision, this problem is often solved by resizing the input image. However, in image forensics this must be definitely avoided not to lose precious details with dramatic impact on performance. As an example, a well-crafted splicing(see the figure) does not show obvious artifacts that allow detection by visual inspection, but suitable analysis tools, like the image noiseprint, expose inconsistencies in the image micro-structure that may be due only to the insertion of alien material in the host image. After strong image resizing, such fine-grain details would be irremediably lost.

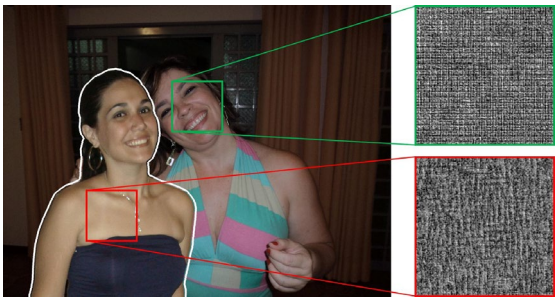


Figure 2: Structural differences in the noiseprint of a pristine and a spliced region.

One can avoid resizing by means of patch-wise processing, and this is the strategy of many state-of-the-art forensic tools that rely on the statistical analysis of local micro-patterns. However, *local* analyses alone are necessarily suboptimal. Clues emerging from the whole image, and at multiple scales, should be combined and processed jointly to make a reliable decision. So, image forensic applications have the need to look, at the same time, at the whole image but also at its tiniest details.

Therefore, we proposed a new framework for full-resolution image forgery detection based on Convolutional Neural Network (CNN) [53]. Our goal was to design CNN-based forensic tools that, overcoming current technological limitations, met the contrasting requirements of full-resolution and full-image training and analysis. We proposed the CNN-based framework depicted in Fig. 3 which makes decisions based on full-resolution information gathered from the whole image. Thanks to gradient check pointing, the framework is trainable end-to-end with limited memory resources and weak (image-level) supervision, allowing for the joint optimization of all parameters. To further boost performance, we use both plain RGB features and noiseprint data, obtained from a pre-trained noiseprint extractor, which is then finetuned together with the whole framework.

4.1.3 GAN fingerprints

GANs are pushing the limits of image manipulation. A skilled individual can easily generate realistic images sampled from a desired distribution or convert original images to fit a new context of interest. With progressive GANs, images of arbitrary resolution can be created, further improving the level of photorealism. Although GAN-based manipulations present often artifacts that raise the suspect of observers, the technology is improving very fast and it is only a matter of time before GAN-generated images will consistently pass visual scrutiny.

In recent years, a large number of methods have been proposed to single out fake visual data, relying on their semantic, physical, or statistical inconsistencies. Statistical-based approaches, in particular, rely

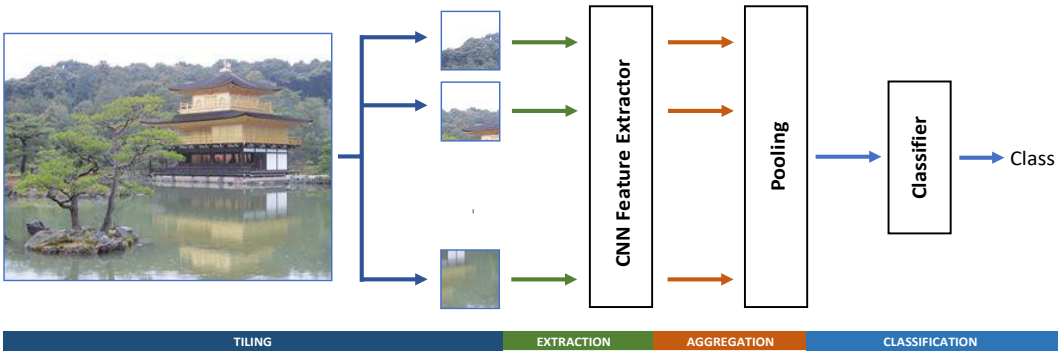


Figure 3: The proposed end-to-end trainable framework.

on the long trail of subtle traces left in each image by the acquisition devices, traces that can be hardly disguised even by a skilled attacker. In fact, each individual device, due to manufacturing imperfections, leaves a unique and stable mark on each acquired photo, the Photo-Response Non-Uniformity (PRNU) pattern, which can be estimated and used as a sort of *device* fingerprint. Likewise, each individual acquisition model, due to its peculiar in-camera processing suite (demosaicking, compression, etc.), leaves further model-related marks on the images, which can be used to extract a *model* fingerprint, like the so-called noiseprints. Such fingerprints can be used to perform image attribution as well as to detect and localize image manipulations, and represent one of the strongest tools in the hands of the forensic analyst.

Of course, GANs have little in common with conventional acquisition devices, and GAN-generated images will not show the same camera-related marks. Nonetheless, they are the outcome of complex processing systems involving a large number of filtering stages, which may well leave their own distinctive marks on output images. Therefore, based on this observation, in this research we set to prove for the first time that each GAN leaves, indeed, its specific fingerprint in the images it generates, just like real- world cameras mark acquired images with traces of their photo-response non-uniformity pattern. By carrying out a process similar to the PRNU extraction, we ended up with stable patterns that characterize all images generated by a specific GAN, and differ from GAN to GAN [52]. Interestingly, not only the network architecture, but also its training impacts on such fingerprints.

Needless to say, GAN fingerprints are very subtle patterns, which cannot be spotted at visual inspection but only through statistical processing. Some insight about them can be gained by observing their spatial autocorrelation, shown in the figure below for two examples. In particular, the strong regular peaks clearly visible in the figure show that not only the filters but also the upsampling processes impact heavily on the formation of the GAN fingerprints, as confirmed by later findings by other researchers.

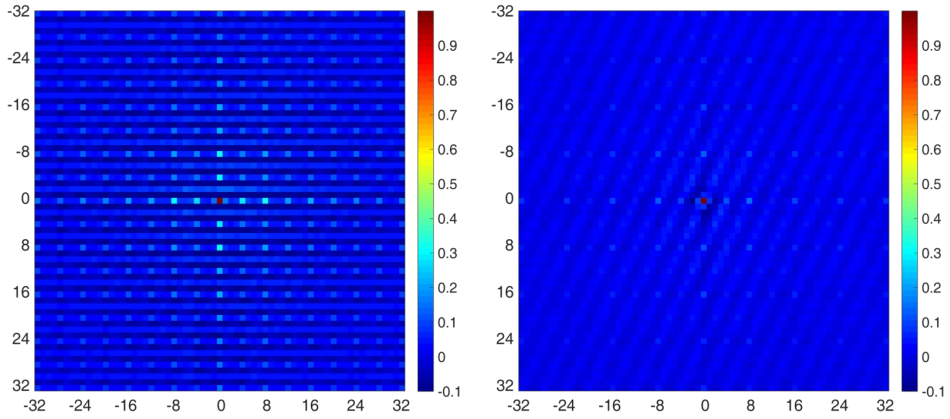


Figure 4: Autocorrelation matrices of the Cycle-GAN and Pro-GAN fingerprints averaged on 512 images.

4.1.4 Video facial manipulation detection

Human faces are by far the most expressive and emotionally-charged pieces of information that circulate on the web. Therefore, advanced AI based methods that are able to modify in a credible way the attributes of faces in videos has raised great alarm. Instead of changing expressions, these methods replace the face of a person with the face of another person. This category is known as face swapping. It became popular with wide-spread consumer-level applications like Snapchat. DeepFakes also performs face swapping, but via deep learning.

To evaluate the effectiveness of a facial manipulation detector, we created a novel dataset of manipulated videos. We considered four automated state-of-the-art face manipulation approaches: two computer graphics-based approaches (Face2Face and FaceSwap) and two learning-based approaches (DeepFakes and NeuralTextures). The Face2Face and NeuralTextures manipulations are facial reenactment methods where the expressions of the source video are transferred to the target video while retaining the identity of the target person. FaceSwap and DeepFakes are instead face swapping methods that replace the face in the target video with the face in the source video.

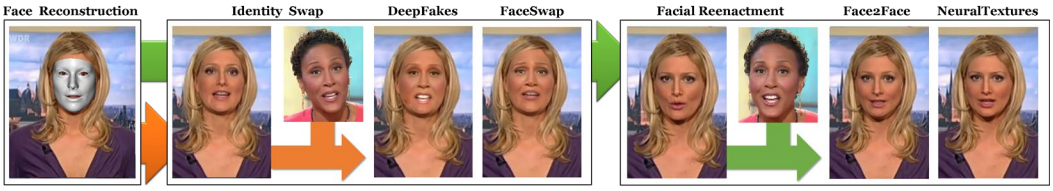


Figure 5: Examples of facial manipulations split in two main categories: identity modification and expression modification.

The dataset, called *FaceForensics++* [70], contains 1000 pristine videos and 1000 forged videos for each manipulation method. To imitate realistic scenarios, we chose to collect videos in the wild, specifically from YouTube. However, early experiments with all manipulation methods showed that the target face had to be nearly front-facing to prevent the manipulation methods from failing or producing strong artifacts. To create a realistic setting for manipulated videos, we generated output videos with different quality levels, similar to the video processing of many social networks. Since raw videos are rarely found on the Internet, we compressed the videos using the H.264 codec, which is widely used by social networks or video-sharing websites. To generate high quality videos, we used a light compression denoted by HQ (constant rate factor parameter equal to 23) which is visually nearly lossless. Low quality videos (LQ) were produced using a factor of 40.

This new large-scale dataset enables us to train a forgery detector for facial image manipulation in a supervised fashion. We cast the forgery detection as a per-frame binary classification problem of the manipulated videos. Since our goal is to detect forgeries of facial imagery, we use additional domain-specific information that we can extract from input sequences. To this end, we use the state-of-the-art face tracking method to track the face in the video and to extract the face region of the image. We use a conservative crop (enlarged by a factor of 1.3) around the center of the tracked face, enclosing the reconstructed face. This incorporation of domain knowledge can improve the overall performance of a forgery detector in comparison to an approach that uses the whole image as input. The extracted region is fed into a learned classification network that outputs the prediction for each processed frame (see fig.6). In order to give a single score for the whole video we fused the results obtained from each frame.

Figure 6: Our domain-specific forgery detection pipeline for facial manipulations.

4.1.5 Video Copy-move Detection and Localization

Video manipulation is becoming more and more widespread nowadays, especially due to the availability of artificial-intelligence tools based on deep learning, which enable even non-expert users to realize deepfake videos. However, video forgery can be carried out also by means of conventional methods, so-called cheap fakes, and may be very difficult to detect and localize. This is certainly the case of video copy-moves, involving the insertion or deletion of compact video objects. By this approach, one can perform both additive copy-moves (e.g., replicating cells in a medical video to modify cell count) and occlusive copy-moves (e.g., pasting background areas to remove a person from a surveillance video). When properly carried out, these attacks can be quite challenging to detect. Indeed, they neither leave traces in the video temporal structure nor insert alien video objects from another video, thus rendering ineffective all detectors looking for statistical anomalies. In addition to this, occlusive copy-moves offer no visual clues or salient keypoints to enable their discovery by visual of automated analyses.

In this research we proposed a new technique for the detection and localization of copy-move video forgeries [18]. In the following, we summarize the main technical approaches used to deal with the inherent problems of this task.

- First, suitable features are computed, invariant to various spatial, temporal (including temporal flipping), and intensity transformations which may be used to disguise the attack. Versions with both original RGB blocks and compact Zernike moments are considered.
- Features are computed densely on a spatio-temporal grid, rather than at salient keypoints. This is an especially qualifying point, since it allows one to deal not only with additive copy-moves (easily detected by key point-based methods) but also with occlusive ones. On the down side, dense-field methods are more computationally demanding than keypoint-based ones.
- A major effort was therefore devoted to limit complexity. First, a nearest-neighbor field (NNF) is built, connecting each feature with its best-matching. To this end, an ad hoc video-oriented version of PatchMatch was developed, exploiting the inherent coherency of the NNF to reduce search complexity. Then, the NNF is processed to single out areas with coherent spatio-temporal displacement as candidate copy-moves. The fast multi-scale processing structure described in the figure is used, with volume of interest detected at the coarsest resolution and then refined.

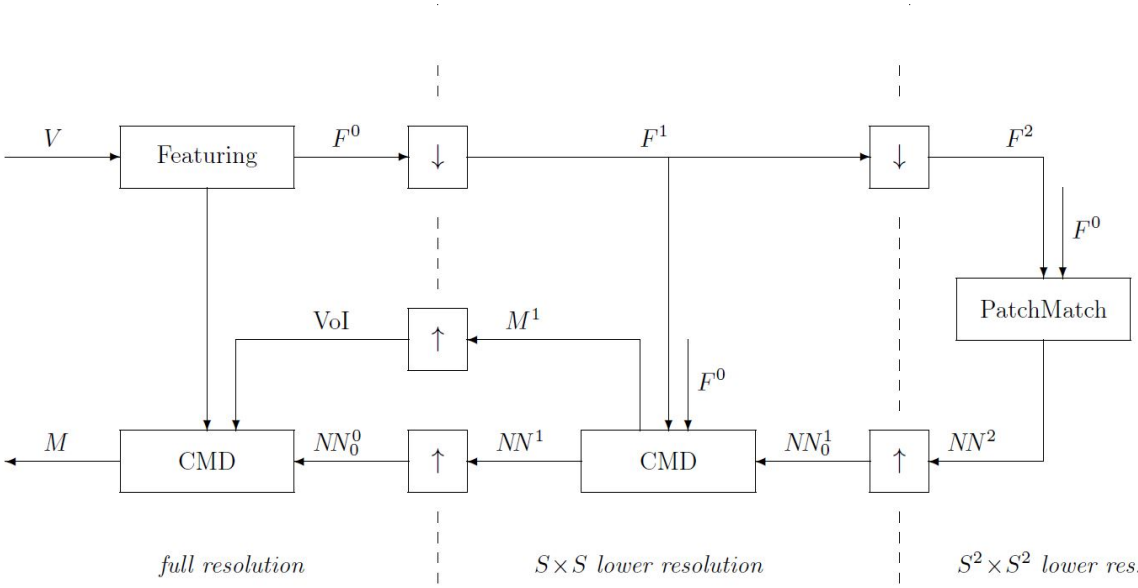


Figure 7: Block diagram of the proposed fast video copy-move detector with multi-resolution processing. The high-resolution field of features F^0 is extracted from the original video, V. This field is then downsampled twice to obtain fields F^1 and F^2. At level 2 (lowest resolution) PatchMatch works on F^2 and F^0 to provide the NN field NN^2. This is upsampled to become the initial NN field at level 1, NN^1. At level 1, the copy-move detector (CMD) works on F^1 and F^0 to refine NN^1 to NN^1, and to extract the detection map M^1 by applying the post-processing. Copy-moved objects are detected in this level, but their shape can be recovered more precisely at level 0. So M^1 is upsampled to define the volume of interest (VoI) and NN^1 is upsampled to become the initial NN field at level 0, NN_0^0. At level 0, the copy-move detector works on F^0, limited only to the VoI, to extract the final output, the detection map $M^0 = M$.

4.1.6 Image Manipulation

We developed the ManTraNet [86], an end-to-end image forgery detection and localization solution, which means it takes a testing image as input, and predicts pixel-level forgery likelihood map as output.

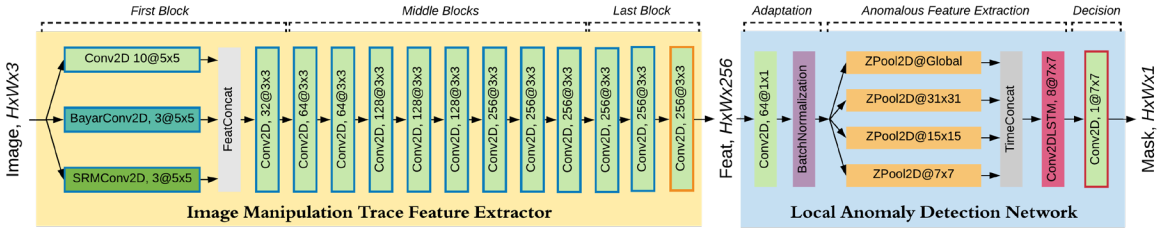


Figure 8: ManTraaNet architecture overview.

The network architecture of ManTraNet is showed in Fig. 8. ManTraNet is composed of two sub-networks as showed below:

- Image Manipulation Trace Feature Extractor: the feature extraction network for the image manipulation classification task, which is sensitive to different manipulation types, and encodes the image manipulation in a patch into a fixed dimension feature vector.
- Local Anomaly Detection Network: the anomaly detection network to compare a local feature against the dominant feature averaged from a local region, whose activation depends on how far a local feature deviates from the reference feature instead of the absolute value of a local feature.

Comparing to existing methods, the proposed ManTraNet has the following advantages:

- Simplicity: ManTraNet needs no extra pre- and/or post-processing Fast: ManTraNet puts all computations in a single network, and accepts an image of arbitrary size.
- Robustness: ManTraNet does not rely on working assumptions other than the local manipulation assumption, i.e. some region in a testing image is modified differently from the rest.

4.1.7 Camera Identification

The acrshortPRNU of a camera is a stable pattern after mild global transformations are applied to a digital image. However, when we extract the acrshortPRNU using the noise residue algorithms, it is still much noisier than the camera fingerprint extracted from flat field images. Our motivation is to research on whether a deep learning model can distinguish between different cameras using the noise residue, and then we can determine how reliable that model is when applied to camera identification and verification.

We adopted a model architecture using the CNN layers from the VGG-16 model followed by several dense layers to make a decision on camera ID: a 256d ReLU-activated dense layer, a 0.5 drop-out layer, a 145d sigmoid-activated dense layer, and a 102d softmax dense layer. 102 is the number of camera classes.

The training set (MFC19) was augmented with image rotated by n x 90 degree (n=1,2,3) and partitioned into training and validation using a 4:1 ratio. We trained the model and measured the 102-class closed-set camera classification accuracy on MFC19. The accuracy on the training partition is 36.2%
and the accuracy on the validation partition is 36.6%.

We find the following three applications of our model to be useful:

- Camera verification: Given tuple <image, hpid>, determine if the image is taking using camera hpid, and produce a confidence score.
- Same-camera image pair verification: Given two images <image 1, image 2>, determine if they are taking using the same camera and produce a confidence score. Instead of letting the model determine the posterior probability for each camera class, each image is represented using the sigmoid layer output as features. And the cosine similarity is computed between features of two images.
- Near-duplicate verification: Use the image pair verification score to verify if two images are near-duplicates

4.1.8 Two-branch Recurrent Network for Isolating Deepfakes in Videos

(a) Expression Transfer + Lip Sync (b) Realistic Face Swapping (c) "Shallow Fakes" or "dumbfakes"

Figure 9: Different types of facial video manipulations. Fake videos can be roughly categorized in three groups of manipulations; the first two groups (a,b) can be referred to as "DeepFakes" since they involve a complex AI model often based on deep neural networks; the last type (c) involves other "shallow" manipulations [4, 79] instead, such as slowing down the video and audio; such effects can still be very effective in misleading the public.

Introduction and Motivation: Social networks and multimedia content improve human connectivity and information sharing. On the other hand, *visual* misinformation and technology-facilitated manipulations have dramatically increased on social networks and the Internet [1]. Nonetheless, image manipulation is not new. Falsification of lithographs or photographs has been used for many years to reinforce political ideas or political characters [29] or to practice censorship by erasing people from pictures. In the modern era of digital pictures, perpetrators used commercial software and "elbow grease" to create realistic swapping of faces given a pair of still images. Although some of these results look very realistic, they involve a huge amount of manual work using a personal computer and an expensive raster graphics editor to produce just a single image [34].

Lately, democratized artificial intelligence (AI) made it very easy to produce highly realistic face swaps with a few clicks, giving the ability to non-experts to synthesize content with "Hollywood-like" quality by simply using off-the-shelf applications and open-source tools [63]. The technology, dubbed "Deepfake" has been quickly developed to process videos, transferring the identity of a subject from a *source* video into a *target* video. Unlike manual digital editing, face swapping in videos became effective and efficient, reaching hyper-realistic results, thanks to recent advances in data synthesis using Generative Adversarial Networks (GAN) [28], Deep Convolutional Neural Networks (DCNN) [45, 44], and AutoEncoders (AE) [41]. It also became easily available to non-experts through customized applications, such as DeepFaceLab [2], or even mobile applications, such as Zao [3].

Research Objective: The research objective of this effort was to develop a DeepFake Detection technology that could operate on videos to detect realistic AI-backed face manipulations of the family of those depicted in Fig. 9(b). The approach operates on the visual content provided by a video and is able to predict which segments of the videos are likely to be manipulated. One of the key issues in developing DeepFake Detection systems is to make them able to generalize across datasets and manipulations types. In this sense, we optimize our method for better generalization across datasets, reaching a good balance between bias and variance [65, 80, 22], i.e., performing remarkably on same dataset used for training [71] yet transferring reasonably well across datasets [47, 21].

Technical Approach: The objective is to learn a classifier for the detection of manipulated faces, squishing a set of aligned video frames[1] $\exists \mathbf{I} \in \mathbb{R}^{H\times W\times 3\times F}$ to an embedding $\Phi(\mathbf{I}) \in \mathbb{R}^{D}$ so that the representations of natural faces are compact around a reference centroid $\mathbf{c}$ and manipulated faces are spread out, ensuring a large margin between tampered and untamperd faces.

In Fig. 10 we introduce a two-branch backbone representation extractor $\Phi(.)$ based on densely connected layers [36]. Φ learns to fuse different representations obtained using regular convolutional filters Φ_{RGB} and representations extracted using multi-scale Laplacian of Gaussian [10] kernels Φ_{LoG}. Φ_{LoG} suppresses the visual information present in the low-level feature maps, effectively acting as a band-pass filter to amplify generation artifacts.

The combined features maps are then fed to the backbone that ends with a bi-directional Long Short-Term Memory (LSTM) for temporal modeling. $\Phi(\mathbf{I})$ indicates the concatenated output from the two bidirectional LSTM streams. The entire recurrent model is supervised through a novel formulation. Unlike recent methods [71, 74] that use classification losses for detection, we introduce a loss function that encourages the compactness of the representations of untampered faces, while distancing the

[1]Throughout this section $\mathbf{I}$ indicates a sequence (or window) of aligned faces from video frames of cardinality F.

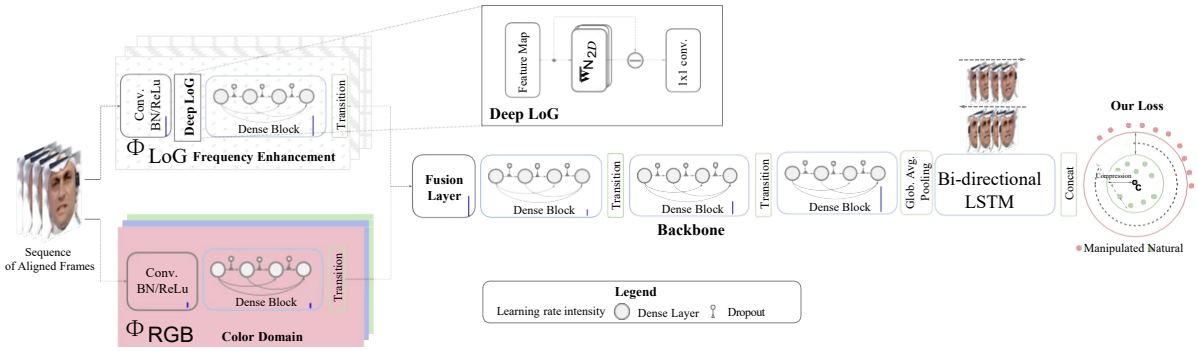
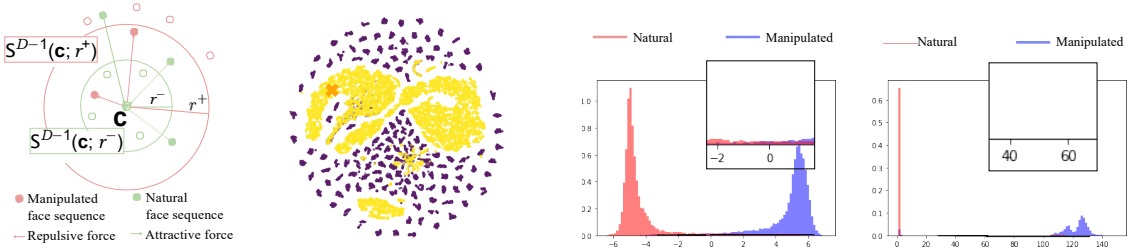


Figure 10: Our video-based face manipulation detection architecture. A face image is processed by two independent DenseBlocks: the first learns to suppress high-level content and amplify a wide range of frequencies using a Deep Laplacian of Gaussian (Deep LoG) layer (frequency enhancement); the second is a classic branch that works in the color domain. The two feature maps are fused so that a backbone of dense blocks learns a rich representation. The architecture uses dropout after each DenseLayer and a different learning rate per layer to mitigate overfitting. Our architecture ends with a bi-directional LSTM layer for video-based modeling and is supervised using a novel loss formulation.



(a) Loss idea **(b)** Feature space **(c)** Logit distribution **(d)** Our distribution

Figure 11: Loss formulation. (a) The loss induces compression of the natural face sequences within a inner hypersphere placing easier samples close to $\mathbf{c}$ and tougher samples at the boundary; meanwhile it induces a large margin forcing the manipulated face sequences outside the outer hypersphere (b) t-SNE [51] visualization of the feature space on the test set between natural faces (yellow) and deepfakes (violet). The center $\mathbf{c}$ is shown as an orange cross. (c) Genuine-Impostor distribution of logits with binary cross-entropy and (d) with our loss function: imposing a wider margin induces less confusion in the distribution.

representations of manipulated faces, for better, wider separation boundaries. At test-time, given an input sequence $\mathbf{I}$, the method obtains the distance $\|\Phi(\mathbf{I}) - \mathbf{c}\|_2$; the larger the distance the higher the likelihood of the sample being manipulated. The formulation of our loss function is explained below.

We propose a new loss for better isolating manipulated faces inspired by recent work on one-class classifiers, such as one-class Deep Support Vector Data Description (Deep SVDD) [73]. The new formulation induces compactness of the embedding space for sequences of unmanipulated faces. However, unlike [73], the proposed loss employs manipulations synthesized by a few generators as negative samples enforcing a larger margin to the natural face sequences.

More formally, we optimize the entire recurrent network defined in Fig. 10 through a cost function that organizes the feature space such that the variability of sequences of natural faces is compacted toward a reference center while the representations of manipulated face sequences are placed far apart at the boundaries of the feature space. Before training, we begin by pre-computing a reference center $\mathbf{c} \in \Re^D$ by averaging the encodings of all the natural, unmanipulated face sequences in the training set. The encodings are obtained by taking the responses of our entire architecture with two-branches and the bi-directional fore training. The concatenated bidirectional structures are extracted using the same network that is pre-trained. The two-branches and the backbone are pre-trained on ImageNet. When the training starts, all the features are aligned to this predefined embedding space. Then we define two hyperspheres centered around $\mathbf{c}$ to constrain the feature space so that natural faces lie within $S^{D-1}(\mathbf{c}; r^-)$, while manipulated faces are kept outside $S^{D-1}(\mathbf{c}; r^+)$. The loss induces compression on the regular faces embeddings. However, unlike [73], we avoid reducing all samples to a single high-dimensional point and mitigate overfitting by requiring compression up to an internal inner margin defined by the radius r^- of the first hypersphere. Furthermore, the proposed loss enforces sequences of manipulated faces to be kept outside the second hypersphere defined by the radius r^+. The loss L given a mini-batch

$\Omega \in \mathsf{R}^{H\times W\times 3\times F\times B}$ of face sequences is defined as shown in Eq. (2):

$$\mathcal{L} = \frac{1}{|\Omega_{\text{nat.}}|}\sum_{i\in\Omega_{\text{nat.}}} \max\left(0, \left\|\Phi(\mathbf{I}_i) - \mathbf{c}\right\|_2 - r^-\right) + \\ + \frac{1}{|\Omega_{\text{man.}}|}\sum_{j\in\Omega_{\text{man.}}} \max\left(0, r^+ - \left\|\Phi(\mathbf{I}_j) - \mathbf{c}\right\|_2\right), \quad (2)$$

where $\Omega_{\text{nat.,man.}}$ selects natural and manipulated face samples, respectively. For this loss to be valid, it has to hold that $0 < r^- < r^+$ and the margin imposed between the two classes is $m = r^+ - r^-$. The values of the two radii have to be set according to the dimensionality D of the feature embedding. The loss mitigates the problem of class imbalances by normalizing each term by its cardinality. Further, the second margin r^+ is essential to the loss because the network may choose to lower the cost just by pushing the negative samples indefinitely, without inducing compression on the natural faces.

Fig. 11a illustrates the basic idea of the proposed loss, and Fig. 11b demonstrates the feature space of the test set of natural faces vs deepfakes. The features are mapped to R^2 using t-SNE optimizing a plain DenseNet model. Natural faces are compressed while manipulated faces lie at the boundaries. The clusters formed by videos are visible for the manipulated faces. Fig. 11c shows the genuine and impostor distribution of the logits at inference time for a model trained for discerning real faces from deepfakes using binary cross-entropy on FaceForensics++ [71]. Although the distribution presents two peaks corresponding to real and deepfakes faces, the variance of those distribution is not minimized, and, more importantly, real face logits are spread out toward the manipulated faces thereby negatively affecting the detection rate at a low false alarm regime. In contrast, Fig. 11d offers the distribution of the distances from the center $\mathbf{c}$ for the two classes. Using the proposed loss we achieved compression of the natural faces and a clear separation from the manipulated faces, visible when zooming in a highly confusing region.

Interpretation: Eq. (2) shares similar traits with the formulation in [73] with a few key differences. First, we have a secondary term for supervision for abnormal cases. Second, we have margins that avoid overfitting and better separate the two classes. The loss function also resembles the classic formulations found in deep metric learning such as contrastive loss functions [83], although in our case the optimization is better constrained since the network is allowed to "move" only $\Phi(\mathbf{I})$ while $\mathbf{c}$ is kept fixed. Finally, we spare the sampling of pairs or even triplets [75] which significantly reduces training complexity. Our loss differs from recent formulations: [82] uses softmax while we do not; it also sets one center for each class while we have a single center for both classes; finally, unlike us, [82] updates the centers while training.

4.2 Physical Integrity

4.2.1 Robust Analysis of the Direction of Incident Light

The direction of incident light was first forensically investigated by Johnson and Farid [38]. It is arguably the most popular physics-based forensic cue. The idea is to compare the lighting environments of two objects under investigation, with the goal of exposing spliced images. However, the classical approach, as well as follow-up works [39, 23, 66] impose strong assumptions on the scene, which severely limits their applicability in practice: 2-D estimation methods require manual annotations of carefully selected contours [38]. 3-D methods are constrained to objects to which a 3-D model can be robustly fitted, which essentially reduces the applicability to the comparison of faces.

One major outcome of this investigation is a novel cue for estimating 2-D lighting environments. The key observation is that the average of all gradients on the surface of a sphere always points in the direction of the light source [67]. We show in our work that this finding can be directly relaxed to mostly convex objects with significant texture [55]. This relaxation yields a forensic cue that is very robust to strong image compression and downsampling, and even to major object segmentation errors. Moreover, partitioning the object surface into sectors extends the analysis of lighting environments slightly beyond 2-D: diverging gradients indicate broad environment light, while converging gradients indicate a light source in front of the scene. These properties are illustrated in Fig. 12. The two images on the top left show a gray input sphere and a textured input sphere. The two images on the top right show that the direction of incident light is robustly estimated with 45° incident angle on the textured sphere, even if only a part of the sphere is segmented. The two left images in the bottom row show that the direction of incident light is consistently and correctly estimated when all gradients are taken into consideration and when subgradients from partitions of the sphere are used. The two images on the bottom right show that

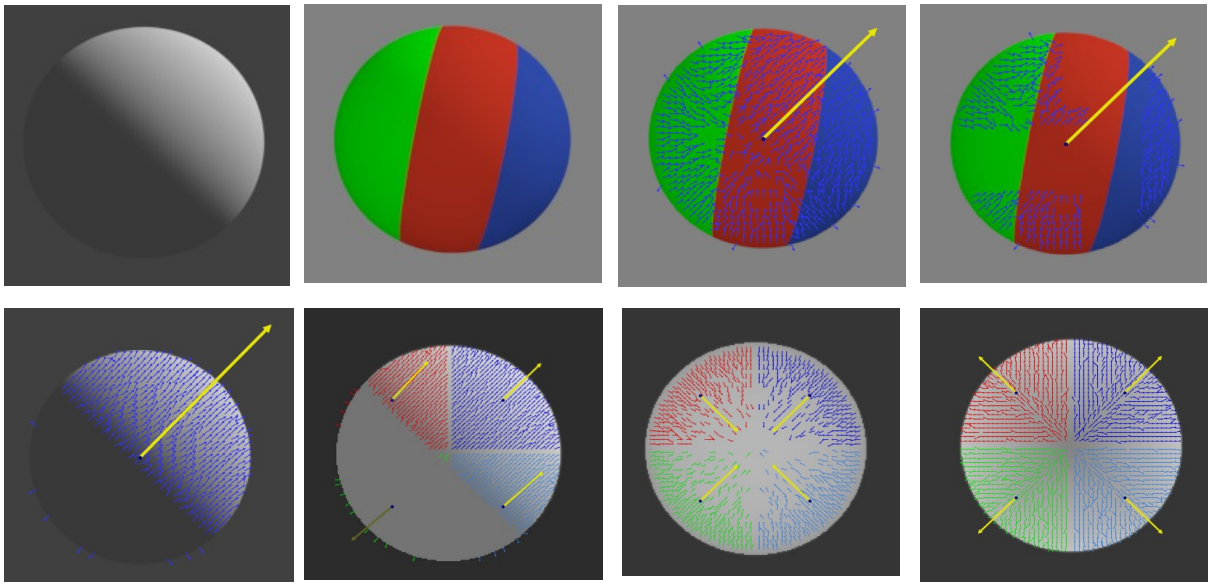


Figure 12: Gradient distribution on a sphere. Top, from left to right: gray input sphere, textured input sphere, illumination direction on full and partially segmented textured input sphere. Bottom, from left to right: illumination direction on gray sphere for one segmentation and four object-local partitions point in the same direction, diverging partition gradients for environment light, and converging partition gradients for light in front of the scene.

environment light leads to diverging subgradients, while light in front of the scene leads to converging subgradients.

The gradient-based method calculates elementary statistical features from the gradient field. These features are classified to compare the lighting environments on pairs of segmented objects of a scene. The main benefit of this method is its remarkable robustness to downsampling, lossy compression, and also to variations in texture and surface reflectance properties, and to segmentation errors. This makes this method quite robust on data that cannot be analyzed with previous lighting-based forensic algorithms.

4.2.2 Segmentation-free Lighting Estimation

One precondition of the gradient-based lighting estimator is to segment the scene into meaningful objects. In many practical scenarios, this is not a serious limitation: powerful state-of-the-art object segmentation algorithms cover dozens or hundreds of object classes. However, the dependency on a segmentation limits the analysis to actual objects in the scene, while unordered structures and background elements are left out.

To include also such unordered structures into the analysis, we also propose a machine-learning based approach to lighting estimation that does not require any object segmentation. The proposed method directly estimates the lighting environment of a rectangular image area.

To this end, we calculate ground truth lighting environments from light probes of more than 1000 scenes from the dataset by Murmann *et al.* [59]. This data is used to train a Convolutional Neural Network (CNN) that regresses for each patch of the image its corresponding lighting environment. To this end, a L_2-loss is calculated on the first nine spherical harmonics coefficients on patches of 150 x 150 pixels. The consistency of lighting environments is assessed in a Siamese architecture that uses the CNN as submodule. A schematic overview of this approach is shown in Fig. 13.

4.2.3 Color Fingerprinting from the Scene and the Camera

The formation of colors in an image depend on the scene reflectance, the spectral distribution of the illuminant, and on camera-internal processing, most notably the color filter sensitivity, camera white-balancing and additional camera-internal nonlinearities like gamma correction. Previous works focused on isolating the color of the illuminant to expose spliced images that were captured under different lighting conditions.

We propose to go one step further, and to include both the color of the illuminant and the camera-internal color processing into a combined descriptor. This offers the possibility to expose spliced images

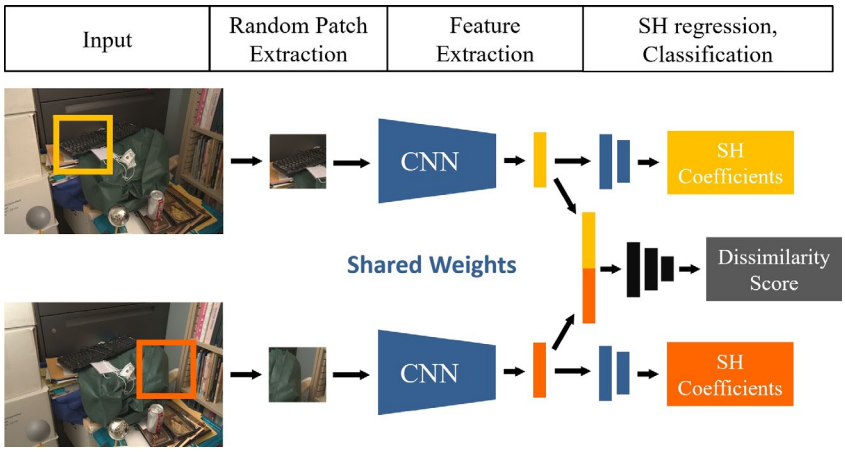


Figure 13: Overview of the learning-based segmentation-less lighting estimation algorithm.

that were either acquired with different cameras, different white-balancing settings, or different scene illumination.

To this end, we propose to learn a metric space into which the features from a rectangular image patch are embedded. This space maps patches that are captured under identical lighting or camera settings to nearby locations, such that their distance in the metric space is small. The challenge here is to learn a descriptor that is invariant to varying object texture.

For this task, we leverage a Residual Neural Network (ResNet) ResNet-50 architecture with an input patch size of 128×128 pixels, pre-trained on ImageNet. A specialized training dataset is used to achieve covariance with the imaging conditions with simultaneous invariance to textures. To this end, we use several datasets of RAW images, i.e., raw sensor readouts that have not been subject to the camera-internal development of the final image. Each raw image is converted to 12 final images using different white-balancing and color-conversion settings available in `LibRaw`. More specifically, each image is white-balanced with each of the modes "autoWB", "cameraWB" and "noWB", combined with each of the four color transformations "raw", "sRGB", "Adobe" and "ProPhoto". The resulting images generally differ in their color appearance. However, for some combinations of scenes and acquisition devices, a subset of the pipelines lead to almost identical results. After removing broken and completely over- or underexposed images, we employ a total of 5997 RAW images from the RAISE database [19], 4998 RAW images from the MIT-Adobe FiveK dataset [11], 1632 RAW images from the dataset by Nam and Kim [60], and 645 RAW images crawled from `raw.pixls.us`. We use the preset training/validation/test split for the dataset by Nam and Kim [60]. The remaining data is split by scenes with a ratio of 0.8, 0.1, 0.1. This yields a total of $|S^{\text{train}}| = 10494$, $|S^{\text{val}}| = 1463$, and $|S^{\text{test}}| = 1315$ scenes for training, validation and test, and thus $|S^{\text{train}}| \cdot |C| = 125928$, $|S^{\text{val}}| \cdot |C| = 17556$, and $|S^{\text{test}}| \cdot |C| = 15780$ images in total.

The desired invariances and covariances are achieved by enforcing two conditions during training. First, the embeddings of patches from the same image must be closer than patches from the same scene but processed with different camera color pipelines. Second, the embeddings of patches from the same image must be closer than pairs from different scenes with arbitrary color pipeline. These constraints focus on differences in the color image formation. Texture differences are suppressed, as they are not subject to the distance constraints. As a side note, these conditions implicitly create an embedding space that can be marginalized in two directions, namely towards the identification of different scene illumination, and towards the identification of different camera settings.

The primary application of this embedding space is to compare pairs of patches for their color consistency. These pairwise comparisons can be aggregated to obtain a method for forgery localization, i.e., that indicates an image area that deviates from the background. Forgery localization is performed by first calculating the medoid of all embeddings, and to mark all patches that significantly deviate from that medoid. It is in principle also possible to perform manipulation detection, i.e., to make a binary statement whether an image is spliced or not, by calculating the average distance across all pairs of patches. Alternatively, the more complex MeanShift aggregation can also be used for detection [12].

4.2.4 Fingerprinting of JPEG Library Chroma Subsampling

The formation of a color image differs also between Joint Photographic Experts Group (JPEG) library implementations. In the JPEG algorithm, the color space is transformed to YCbCr. Here, Y denotes luminosity, and Cb and Cr denote the luminosity-normalized blue and red chroma components. Since the eye is less sensitive to color variations, the Cb and Cr channels are typically spatially subsampled prior to encoding, to further reduce the storage size of the image.

This subsampling step is performed differently across JPEG libraries. In particular, `libjpeg-turbo`, which is a widely distributed fork of the standard library `libJPEG`, introduces during subsampling a small periodic offset in horizontal direction of each 8 x 8 pixels JPEG block.

This periodic offset is invisible to the eye, and can only be measured in JPEG images with a compression quality that well preserves high-frequency coefficients. Detecting this artifact is relatively straightforward, by correlating each JPEG block with a fixed template that contains the artifact. That way, it is possible to distinguish images that were created with different JPEG libraries. Additionally, it is also possible to expose local manipulations in an image. This refers either to local editing like inpainting, where the structure of the artifact is locally destroyed, or splicing if images from two different JPEG libaries are combined. However, since the artifact critically depends on the high-frequency content of an image, it can only be detected at quality levels of about 85 and beyond if it has been introduced in the last compression step, or at quality levels of 97 and beyond if the image has undergone another JPEG compression after introduction of the artifact.

4.2.5 Fingerprinting of Depth Image Calculation in Cameras

Top-of-the-line smartphones have the capability to calculate a depth image of the scene, with applications in image enhancement, image segmentation, and biometric authentification. Moreover, the depth image is typically silently embedded into the camera JPEG image to facilitate post processing of an image after acquisition. When downloading an image from a camera, the depth image can be directly accessed.

There are different possibilities for the hardware setup that is used to obtain such a depth image, e.g., by using time-of-flight sensors, stereo cameras, or by estimating depth from a monocular sensor. There are also different possibilities for the algorithmic transformation of a hardware measurement to the final depth image.

This large space of possibilities offers new, interesting perspectives for a forensic analysis. First, the actual realization of the depth image can be used as a fingerprint for the acquisition device. Second, if an image is manipulated, its accompanying depth image must either be removed from the JPEG container to prevent detection, or it must be manipulated in a consistent way.

We make the assumption that the depth image is available, and investigate the possibilities to predict from a patch of the depth image the acquisition device.

To this end, we use a total of twelve monocular and stereo algorithms for depth-estimation, and apply these algorithms on the 7481 RGB stereo image pairs of the KITTI dataset. Distinguishing these algorithms serves as a proxy task for training, since the actual implementations of smartphone manufacturers are not publicly available.

The depth images are used to set up two training tasks. First, we train a deep neural network with twelve output neurons for algorithm fingerprinting. Second, we also use this fingerprinting CNN also in a Siamese network to distinguish whether two patches stem from the same depth estimation algorithm or not. Both setups are illustrated in Fig. 14. For the CNN architecture, we evaluate architectures that are also commonly used for other forensic tasks, namely Extreme Inception Neural Network (Xception), ResNet-50, Multimedia and Information Security Lab Neural Network (MISLNet), and Mesoscopic Anal- ysis Neural Network (MesoNet). For application on real smartphone data, these pre-trained networks are refined with few-shot tuning on three images per smartphone device.

4.3 Semantic Integrity

4.3.1 Provenance Filtering

Fig. 15 shows the overall architecture of the developed provenance detection system. As shown in Fig. 15 (a) base detection is an initial stage to detect all images created using the same base image. Starting from a cluster of similar images (e.g., illustrated in Fig. 15 (b)), splice detection is performed by subtracting the base image from the image containing the spliced object. A trimming step is applied to reduce the number of subtractions (illustrated as solid arrows in Fig. 15 (c)). In the donor detection

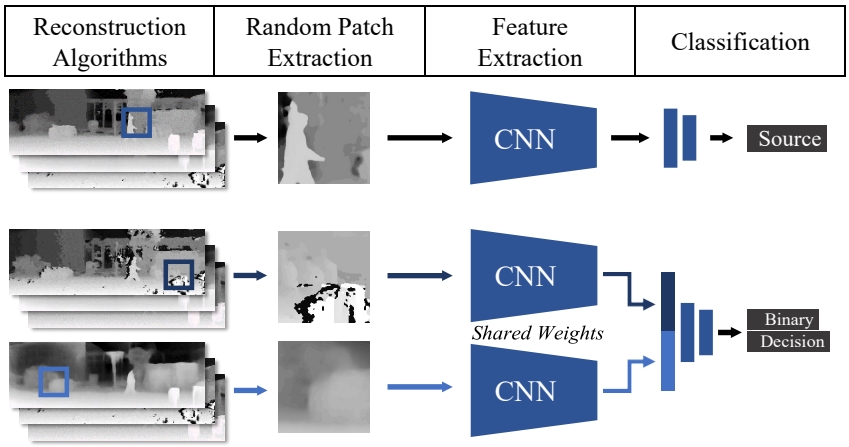


Figure 14: Overview on the depth map fingerprinting pipeline. Top: a single CNN is used for depth map fingerprinting. Bottom: a Siamese architecture of this CNN is used for patch-wise comparison of depth-map consistency.

stage, spliced objects are used as queries to search for donor images that have these spliced objects (illustrated as red boxes in Fig. 15 (d)). In both base detection and donor detection stages, the cosine similarity is measured using the VGG-16 [11] features. Since the extraction of VGG-16 features is still not fast enough, the intensity of the down-sampled version of the original image (down-sampled into 24 by 24 pixels) is also used as a rapid feature extraction step to quickly determine if two images are near-duplicates. Near-duplicates are likely to be images created using the same base image. For a pair of feature vectors, the cosine similarity is computed to measure their similarity.

In the donor detection stage, the scores for two special types of provenances are also computed. The first type is the probe-as-donor scores. Probe-as-donor means the probe is a donor rather than a manipulated image. The second type is indirect provenance. When three images are involved in splicing where an object from image A was taken and spliced into image B, and another object from image B was taken and spliced into image C, images A and C became indirect provenances.

Our system produces multiple groups of base and donor detection results, each from a different method. These results were aggregated by taking the top candidates of each method, and merged into the final list of results.

5 Results and Discussion

5.1 Digital Integrity

5.1.1 Noiseprint

Since an image noiseprint is a camera model-related piece of information extracted from a test image, it can be used to perform a whole range of diverse forensic tasks. Therefore, there is a large number of potential applications, some of which have been pointed out and highlighted in the original paper.

Among the many applications, however, image forgery localization is one of the most interesting, both for its intrinsic importance in multimedia forensic analysis, and for its good match with the characteristics of noiseprint themselves. Therefore, focusing on this task, we implemented a forgery localization algorithm, inspired to Splicebuster, but based on the image noiseprints as original data. Then, we carried out experiments on a large number of datasets currently used in the community. Some of them focus only on splicing, like DSO-1, VIPP, and the FaceSwap dataset. Others are much more challenging, and present a wide variety of manipulations, sometimes cascaded on one another on the same image. This applies in particular to the datasets designed by NIST for algorithm development and evaluation in the context of the Medifor program, which can be considered very challenging benchmarks for all methods under test. As for reference methods, we considered all the most popular and promising proposed in the literature, which can be roughly grouped in three classes according to the features they exploit: compression artifacts, color filter array artifacts and inconsistencies in the spatial distribution of features.

Results are reported in the table below in terms of the Matthews Correlation Coefficient. To ensure a meaningful comparison across datasets so diverse we also report the performance ranking. Best results

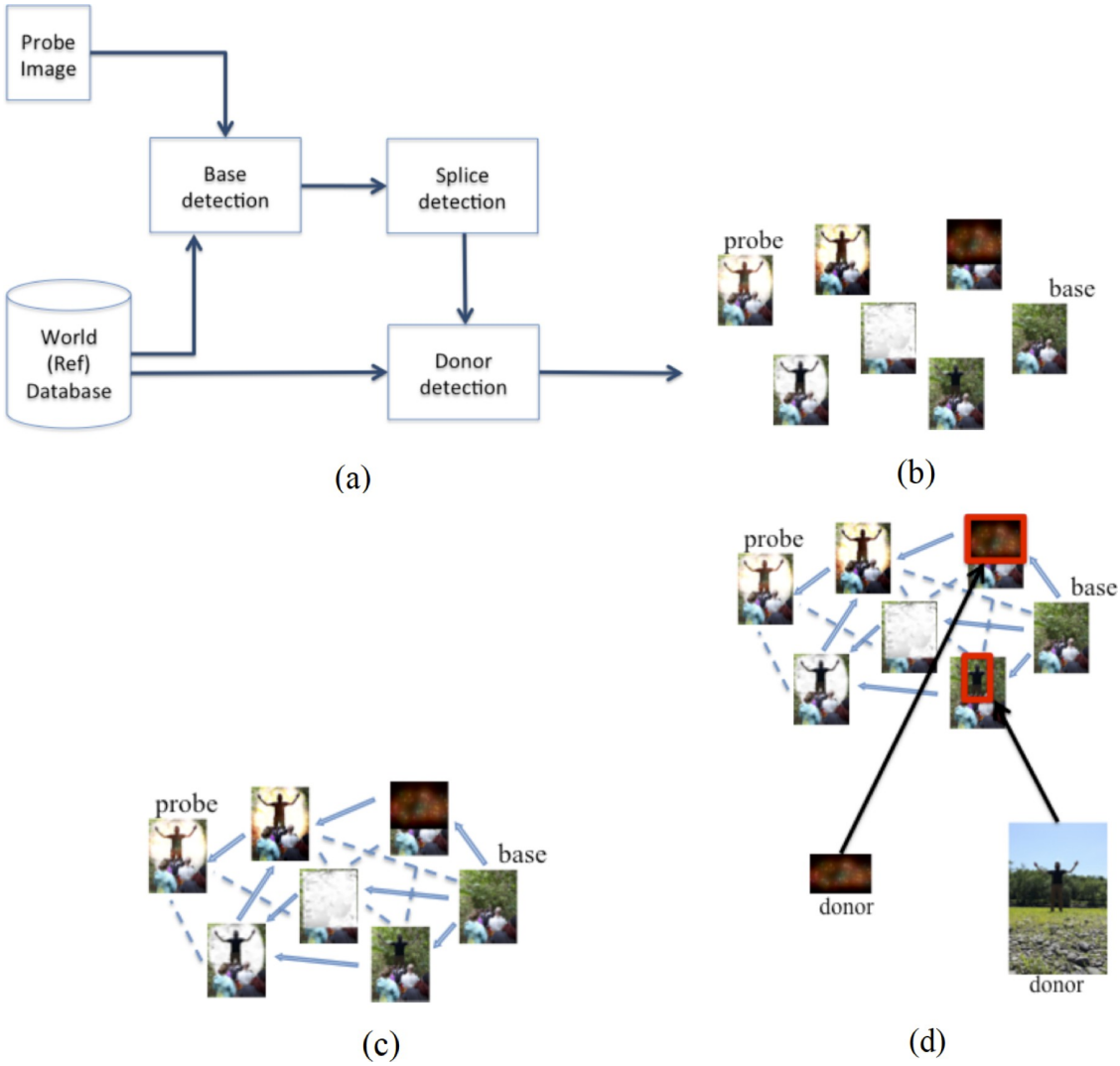


Figure 15: Provenance filtering system diagram.

are highlighted in red. It clearly appears that noiseprint provides the best performance on the average, ranking always first or second among all techniques. Finally, in fig.1 we report just a few examples with forged images, their associated ground truth, and the output of the noiseprint-based algorithm.

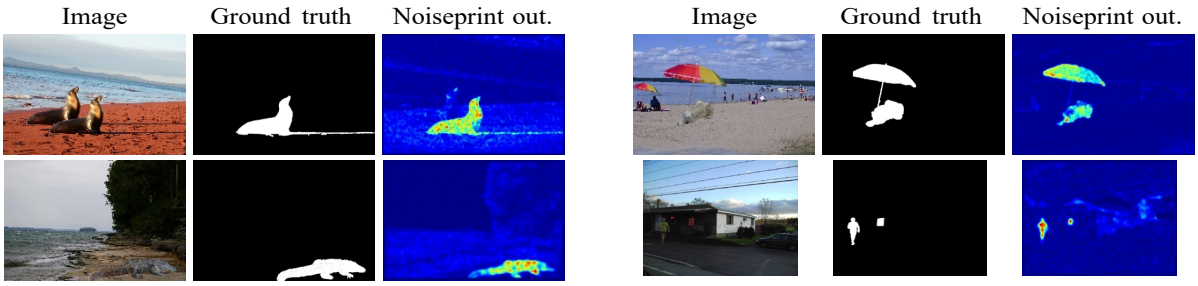


Figure 16: Examples of forged images with relative ground truth and the output of the noiseprint-based algorithm.

5.1.2 An end-to-end trainable approach for image forgery detection

To train the proposed framework we generated a suitable synthetic dataset. Background images are taken from the Vision dataset for camera model identification. To generate manipulated images, we

Table 1: Experimental results on Localization in terms of Matthews Correlation Coefficient

Dataset	DSO-1	VIPP	FaceSwap	Nim.16	Nim.17d2	Nim.17ev	MFC18d1	MFC18ev	AVERAGE
ELA	0.149 (14)	0.190 (12)	0.087 (11)	0.145 (14)	0.103 (14)	0.112 (14)	0.110 (13)	0.115 (14)	0.122 (13.2)
BLK	0.388 (7)	0.365 (8)	0.118 (10)	0.204 (9)	0.163 (10)	0.156 (9)	0.167 (9)	0.153 (11)	0.207 (9.3)
DCT	0.234 (10)	0.376 (7)	0.194 (8)	0.195 (10)	0.154 (12)	0.151 (10)	0.153 (10)	0.159 (10)	0.193 (9.9)
NADQ	0.065 (15)	0.162 (14)	0.040 (15)	0.154 (13)	0.103 (15)	0.113 (13)	0.104 (14)	0.123 (13)	0.101 (14.1)
ADQ1	0.321 (9)	0.473 (3)	0.311 (4)	0.262 (7)	0.181 (8)	0.193 (7)	0.203 (8)	0.194 (8)	0.256 (7.1)
ADQ2	0.464 (6)	0.557 (1)	0.463 (1)	0.305 (4)	0.205 (7)	0.190 (8)	0.299 (3)	0.237 (4)	0.325 (4.8)
CAGI	0.488 (4)	0.429 (4)	0.205 (7)	0.279 (6)	0.242 (4)	0.258 (4)	0.232 (6)	0.215 (5)	0.286 (5.1)
CFA1	0.179 (11)	0.225 (11)	0.072 (12)	0.185 (11)	0.175 (9)	0.148 (11)	0.140 (11)	0.164 (9)	0.193 (9.6)
CFA2	0.168 (12)	0.167 (13)	0.071 (13)	0.184 (12)	0.160 (11)	0.132 (12)	0.136 (12)	0.153 (12)	0.159 (11.3)
NOI1	0.332 (8)	0.276 (10)	0.145 (9)	0.235 (8)	0.226 (5)	0.214 (5)	0.215 (7)	0.212 (7)	0.231 (7.3)
NOI4	0.160 (13)	0.160 (15)	0.052 (14)	0.133 (15)	0.112 (13)	0.111 (15)	0.104 (15)	0.104 (15)	0.113 (14.3)
NOI2	0.487 (5)	0.339 (9)	0.221 (6)	0.296 (5)	0.218 (6)	0.199 (6)	0.251 (5)	0.213 (6)	0.271 (6.2)
EXIF-SC	0.529 (3)	0.402 (5)	0.306 (5)	0.344 (2)	0.320 (3)	0.297 (1)	0.261 (4)	0.260 (3)	0.333 (3.3)
Spliceb.	0.615 (2)	0.391 (6)	0.350 (3)	0.344 (3)	0.328 (1)	0.280 (3)	0.305 (2)	0.281 (2)	0.365 (2.7)
Noiseprint	0.758 (1)	0.532 (2)	0.356 (2)	0.387 (1)	0.324 (2)	0.295 (2)	0.334 (1)	0.292 (1)	0.403 (1.7)

spliced on them objects drawn from a set of 81 objects manually cropped from the uncompressed images of the UCID dataset.

For performance assessment we used a number of datasets widespread in the forensics community with markedly different characteristics. First, the Dresden/FAU synthetic dataset, built like the training dataset but from different sources. Then, the DSO-1 and Korus datasets, where only splicings and copy-moves are present. Finally the very challenging NC2017, MFC2018, and MFC2019 datasets, developed by NIST the context of the Medifor initiative, where images have been manually doctored, often with multiple and possibly overlapping manipulations of various types.

Table 2: Results of all versions of E2E and all references methods on the test datasets. No fine-tuning.

Method	supervision	Dres./FAU	DSO-1	Korus	NC2017	MFC2018	MFC2019	average
Xcep.-resize	weak	0.609	0.539	0.527	0.513	0.570	0.516	0.546
Xcep.-patchwise	strong	0.721	0.643	0.533	0.729	0.711	0.632	0.661
Xcep.-pooling	strong	0.839	0.702	0.561	0.751	0.635	0.633	0.687
SPAM+SVM	weak	0.506	0.768	0.502	0.767	0.631	0.634	0.635
CNN+SVM	strong	0.593	0.728	0.568	0.798	0.702	0.679	0.678
LSTM-EnDec	strong	0.543	0.590	0.521	0.504	0.535	0.542	0.539
ManTraNet	strong	n/a	0.874	0.555	*0.612	*0.758	*0.580	0.676
CFA	–	0.507	0.584	0.598	0.593	0.539	0.526	0.558
DCT	–	0.505	0.614	0.501	0.683	0.523	0.509	0.556
NOI	–	0.558	0.543	0.507	0.678	0.523	0.726	0.589
Noiseprint	–	0.611	0.821	0.583	0.746	0.684	0.662	0.684
EXIF-SC	–	0.599	0.721	0.496	0.709	0.670	0.655	0.642
E2E-RGB	weak	0.958	0.596	0.607	0.774	0.760	0.737	0.739
E2E-NP	weak	0.874	**0.924**	**0.665**	0.766	0.776	0.741	0.791
E2E-RGB+NP	weak	0.914	0.790	0.619	0.762	0.765	0.765	0.769
E2E-Fusion	weak	**0.993**	0.824	0.655	**0.846**	**0.838**	**0.787**	**0.824**

ManTraNet results marked with an asterisk are obtained on approximately 20% of the dataset (small images).

As reference, we considered three natural baselines, Xception-resize, and Xception-patchwise, implementing the extreme options, and Xception-pooling, identical to our proposal except for the isolated rather than joint training of blocks. In addition we used a number of state of the art methods proposed in the literature working both at the image level and at patch-level. All these references are classified in terms of the level of supervision they require, strong, weak (like our proposal), or none.

In table 2 we report the detection AUC for all methods on all test datasets. In the upper part of

the table we group all reference methods, including the baselines, and in the lower part all version of the proposed method with end-to-end (E2E) training. Best results are highlighted in red for reference methods and in blue for our proposal. It results that the best proposed method is always better than the best reference, often with a large margin. Looking at average results, all E2E methods are better than the reference methods, and the best version, E2E-fusion, outperforms the best reference, Xception-pooling, by 0.137. This performance is confirmed by ROC curves, not reported for brevity, and remains the same in all internal Medifor tests.

5.1.3 GAN fingerprints

We carried out a number of experiments to assess the potential of GAN fingerprints for typical tasks in multimedia forensics. Preliminarily, we collected a number of noise residuals, obtained by filtering GAN-generated images and taking their high-pass content. Then, we computed the correlation between noise residuals and GAN fingerprints. In the figure below we show (only for two GANs for brevity) some histograms of same-GAN (green) and cross-GAN (red) correlations. Cross-GAN correlations are evenly distributed around zero, indicating no correlation between generated images and the unrelated fingerprint while, and well separated by same-GAN correlations, almost always positive.

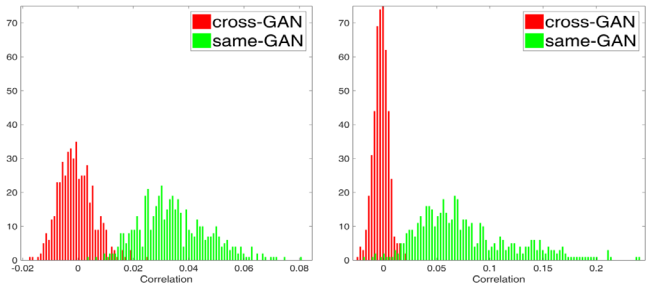


Figure 17: Correlation of Cycle-GAN (left) and Pro-GAN (right) residuals with same/cross-GAN fingerprints.

Then, we tested the use of GAN fingerprints for source identification. we considered three GAN architectures, with different training datasets, since both architecture and training data contribute to the GAN fingerprint.

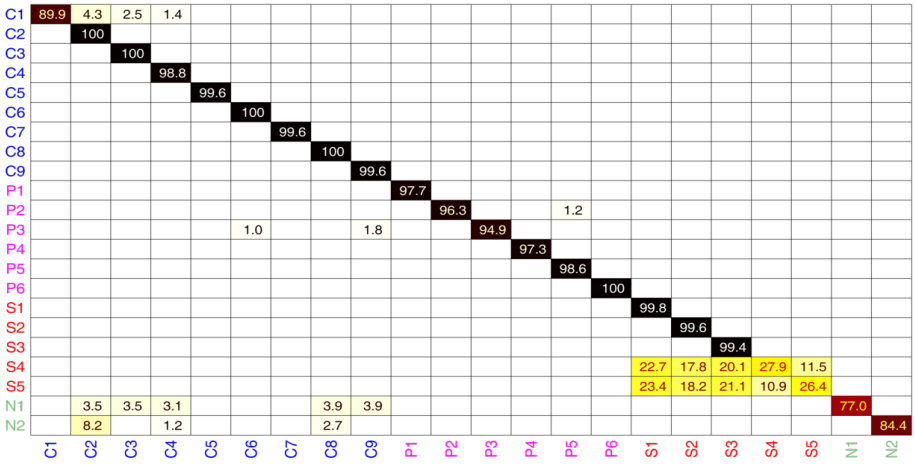


Figure 18: Source identification confusion matrix. Entries below 1% are canceled.

For Cycle-GAN, we considered 9 different image-to-image translation tasks (apple2orange, horse2zebra, monet2photo, orange2apple, photo2Cezanne, photo2Monet, photo2Ukiyoe, photo2VanGogh, zebra2horse), for Progressive GAN 6 different datasets (bedroom, bridge, church, kitchen, tower, celebA), and 5 attributes for Star-GAN (black hair, blond hair, brown hair, male, smiling). Together with GAN generated images, we also considered images acquired by two real cameras. We then performed GAN/camera attribution. For each image, we computed the distance between the corresponding residual and all fingerprints, attributing the image with a minimum-distance rule. Results are summarized in the confusion

matrix below, and show almost perfect attribution. The only exception concerns some Star-GAN images, since the weights of a single generator are shared among all target domains, inducing a significant cross-GAN correlation. A slightly worse attribution performance is observed also for the real cameras, characterized by a lower-energy PRNU.

Finally, we note that this approach was used also in the "GAN Challenge" organized in June-July 2018 by the US NIST in the context of the Medifor program. The goal was to classify as real or GAN-generated 1000 images of widely different resolution, from 52 x 256 to 4608 x 3072 pixels. As baseline method we used a deep network trained on a large number of images retrieved from the InterNet. However, we also tested the GAN fingerprint idea, which allowed us to improve the deep net accuracy by a simple fusion rule, for a final 0.999 AUC.

5.1.4 Video facial manipulation detection

Using our dataset of facial manipulations, we evaluated different state-of-the-art classification methods. We considered a hand-crafted based approach and five network architectures known from the literature to solve the classification task: Steganalysis Features + SVM, Cozzolino et al., Bayar and Stamm, Rahmouni et al., MesoNet and XceptionNet.

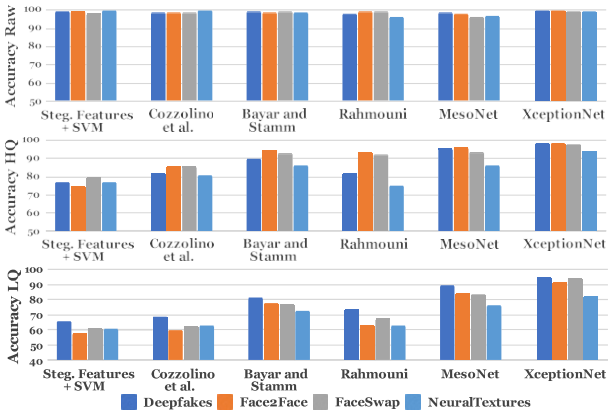


Figure 19: Binary detection accuracy of all evaluated architectures on different manipulation methods.

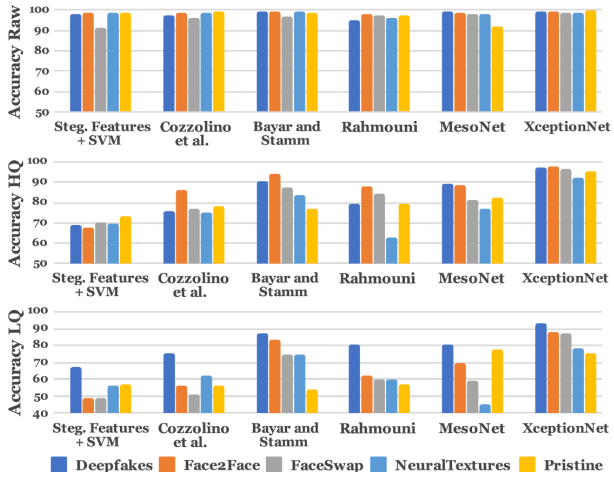


Figure 20: Binary precision values of our baselines when trained on all four manipulation methods simultaneously.

Fig.19 shows the results of a binary forgery detection task using all network architectures evaluated separately on all four manipulation methods and at different video quality levels. All approaches achieve very high performance on raw input data. Performance drops for compressed videos, particularly for hand-crafted features and for shallow CNN architectures. The neural networks are better at handling these situations, with XceptionNet able to achieve compelling results on weak compression while

still maintaining reasonable performance on low quality images, as it benefits from its pre-training on ImageNet as well as larger network capacity.

We also tested the detection variants on a dataset containing images from all manipulation methods. Fig.20 show the results on the full dataset. The experiments highlight that all detection approaches achieve a lower accuracy on the GAN-based NeuralTextures approach. NeuralTextures is training a unique model for every manipulation which results in a higher variation of possible artifacts. While DeepFakes is also training one model per manipulation, it uses a fixed post-processing pipeline similar to the computer-based manipulation methods and thus has consistent artifacts.

5.1.5 Video Copy-move Detection and Localization

To assess the performance of the proposed method we prepared a dataset, called the GRIP dataset from now on, comprising 15 short videos with rigid copy-moves, 10 additive and 5 occlusive. Copy- moves were carried out using After Effects Pro, and show little or no artifacts, just as with a real-world skilled attacker. Inserted objects may be subject to rotation and temporal flipping, moreover, the whole videos may be compressed at various quality factors. All copy-moved videos are available online at http://www.grip.unina.it/ together with their pristine versions and the ground truths. In addition, we used the REWIND dataset available online. This latter dataset, however, comprises only rigid additive copy-moves and comes without a ground truth allowing only for limited analyses.

			Basic 2D			Basic 3D			Fast 2D			Fast 3D		
dataset	case	# videos	det.	f.a.	F	det.	f.a.	F	det.	f.a.	F	det.	f.a.	F
GRIP	plain	15	15	2	0.83	15	1	0.76	14	3	0.79	15	1	0.75
GRIP	QF = 10	15	15	1	0.84	15	1	0.77	14	2	0.74	14	1	0.75
	QF = 15		15	1	0.76	15	1	0.72	13	2	0.65	15	1	0.70
	QF = 20		12	1	0.54	12	1	0.56	13	2	0.53	12	0	0.52
GRIP	$\theta = 5^o$	8	8	–	0.81	7	–	0.73	5	–	0.40	7	–	0.68
	$\theta = 25^o$		7	–	0.71	4	–	0.60	3	–	0.25	4	–	0.44
	$\theta = 45^o$		5	–	0.56	4	–	0.43	2	–	0.12	4	–	0.43
GRIP	flipping	9	8	–	0.81	9	–	0.76	6	–	0.59	7	–	0.59
REWIND	plain	10	8	4	–	9	4	–	8	4	–	6	1	–

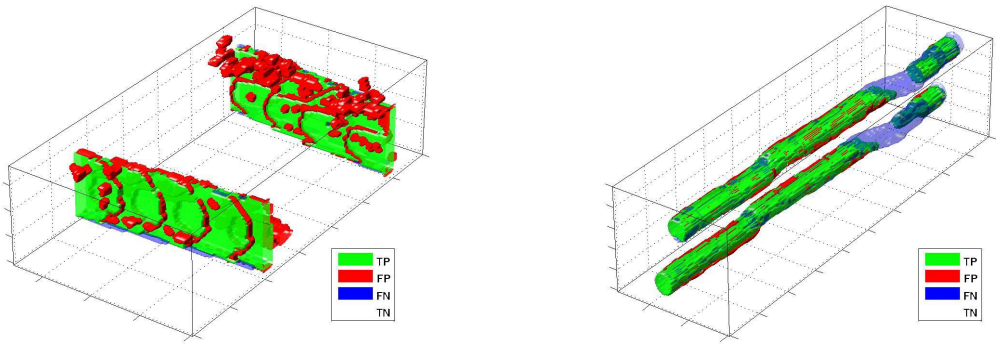
Experimental results are reported in the Table above for various version of the algorithm, Basic/Fast working on 2D/3D features. Detection performance is measured by the number of detected attacks and the number of false alarms (on pristine videos) while localization performance is given in terms of F-measure. The detection performance is near-perfect for the slower versions of the algorithm, and very good also for the faster versions, except in the presence of large rotation angles and strong compression. It is worth underlining that the performance is equally good for additive and occlusive copy-moves, while keypoint-based methods fail on all occlusive attacks. Localization performance is also very good in general, provided the attack is detected. This is also confirmed by the sample detection maps shown in the figure, obtained with the proposed algorithm (basic, 2D features) on some GRIP videos with copy- moves and various operating conditions. Finally, the various solutions proposed to limit computational complexity allow for a huge reduction of running time w.r.t. linear search. This enable laboratory video analysis, but not yet real-time analysis or mass screening of video repositories.

5.1.6 Image Manipulation

A few examples of manipulation detection heat maps using ManTraNet are showed in Fig. 22. The pristine images came from the Dresden Image Database. The three columns are pristine images, manipulated images, and the heat maps, respectively.

The manipulation detection performance (AUC) of the ManTraNet is showed in Table 3. For details about the datasets and experiment settings, see [86].

Methods NIST Columbia COVERAGE CASIA Forgery Types , , , , ELA [29] 0EOI1 [34] 0CFA1 [22] 0J-LSTM [7] 72RGB-N [55] 72ManTra-Net 0Table 8. (training scores and training)



Copy-move with flipping Copy-move with 45^o rotation

Figure 21: Sample color-coded detection maps for videos of the GRIP dataset. True Negative pixels are transparent for correct visualization, True Positive are in green. Only rotation (right) causes the loss of a copy moved segment, following sudden camera motion.

Table 3: (training %, AUC) performance comparisons.

Methods	NIST	Columbia	COVERAGE	CASIA
Forgery Types	splicing copy-move removal	splicing	copy-move	splicing copy-move enhancement
ELA	0% 42.9%	0% 58.1%	0% 58.3%	0% 61.3%
EOI1	0% 48.7%	0% 54.6%	0% 58.7%	0% 61.2%
CFA1	0% 50.1%	0% 72.0%	0% 48.5%	0% 52.2%
J-LSTM	72% 76.4%	N/a	75% 61.4%	N/a
RGB-N	72% 93.7%	0% 85.8%	75% 81.7%	85% 79.5%
ManTraNet	0% 79.5%	0% 82.4%	0% 81.9%	0% 81.7%

5.1.7 Camera Identification

Result on the MFC19 dataset We experimented with camera verification, image pair verification and near-duplicate detection on the MFC19 dataset.

The ROC curve for the camera verification result on MFC19 Eval set is shown in Fig. 23. The corresponding AUC is 85.5%. The best AUC from all teams in the 2019 evaluation was 79.7%.

The ROC curve for the image pair verification result on MFC19 Eval set is shown in Fig. 24. The corresponding AUC is 77.5%. This performance showed us even without providing the camera ID, verification of whether two images were captured using the same camera can still be done with very good performance.

Using the same features we used for image pair verification, we also measured the near-duplicate verification performance on MFC19 Eval set. We labeled all near-duplicate pairs as the ground-truth. The ROC curve is showed in Fig. 25. The corresponding AUC is 100%. Given the good performance, this can potentially be very useful for base detection in provenance filtering.

5.1.8 Two-branch Recurrent Network for Isolating Deepfakes in Videos

Benchmarks and metrics: Experiments are conducted on (1) FaceForensics++ [71] (FF++), (2) Celeb-DF [48], (3) and the Deepfake Detection Challenge Preview Dataset [21] (DFDC) . We report results at the video-level and also at the frame-level. Given that our method works at a sequence level, when comparing to other methods, we made sure that the number of samples prior computing the ROC is the same for all methods when comparing at the frame-level or, at least, that that all methods observed the same quantity of data. Further, we use standard metrics such as True Acceptance Rate (TAR) at low False Acceptance Rates (FAR), similar to [42, 77]. Besides standard area under receiver operating curve (AUC), we further use global metrics yet at a low false alarm rate such. These metrics can shed light on

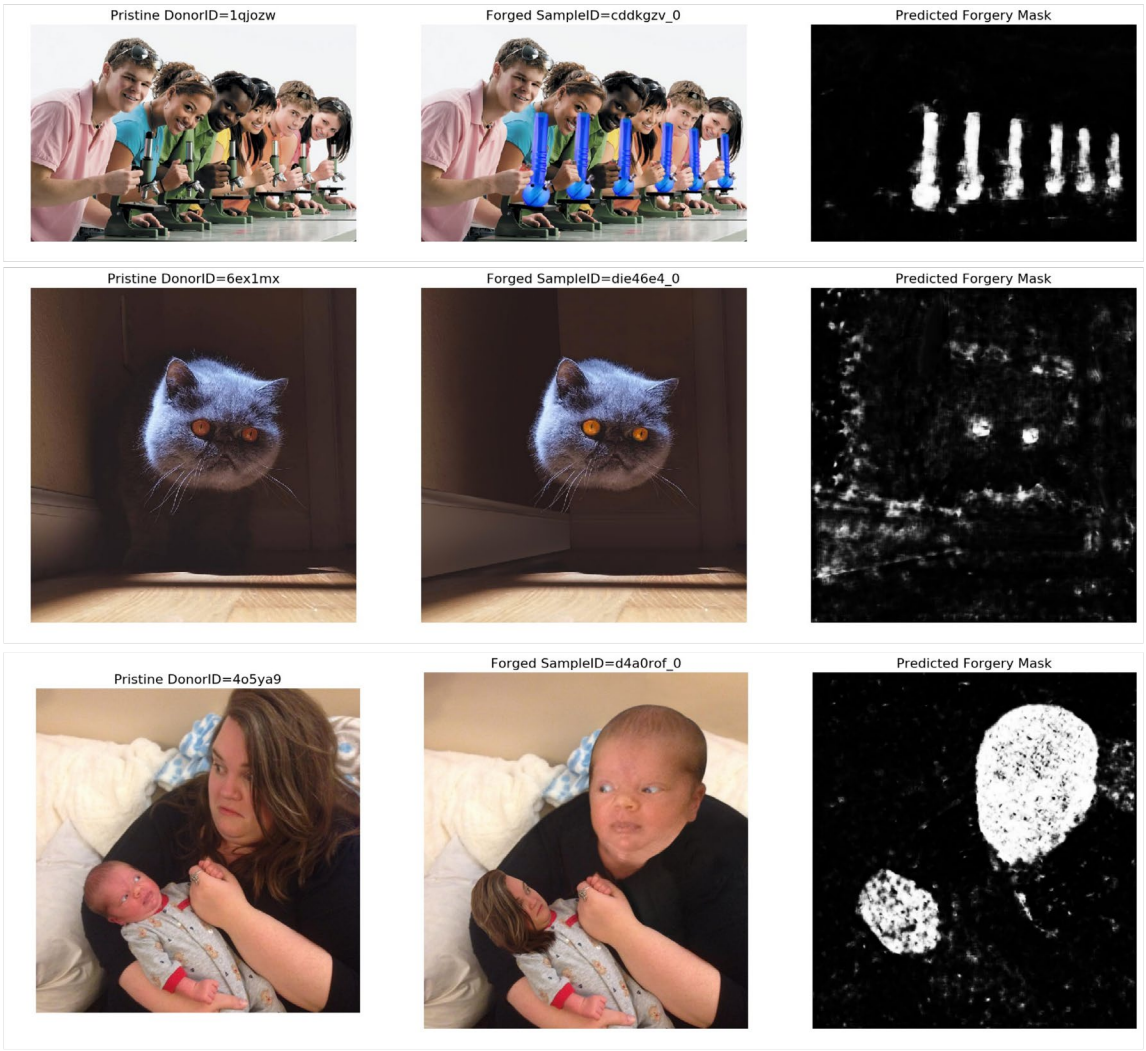


Figure 22: ManTraNet manipulation detection heat maps output (Dresden DB).

performance in realistic operational scenarios, thereby requiring detectors to operate at a very low false alarm rate and raising the bar for the community. We used the standardized partial AUC or pAUC [58] and our tAUC, that is defined as AUC yet taking into consideration only the low false alarm rate up to a cut-off point FAR_τ, thereby ignoring high false alarm rates. tAUC is computed as the ratio between the area of TARs up to a given low FAR_τ normalized by the total area up to the FAR_τ value. Given $\mathsf{F}_\tau = \{0, \ldots, \text{FAR}_\tau\}$, then tAUC at an operating point τ is defined as $\text{tAUC}_\tau \doteq \frac{\sum_{i \in \mathsf{F}_\tau} \text{TAR}_i}{|\mathsf{F}_\tau|}$.

Implementation and Hyper-Parameters: Unless otherwise stated, we used the following settings. The global learning rate μ is 1e-03 using the Adam optimizer and the results are produced with LSTM. The learning rate is decreased three times by a factor of 10. We decrease it every time the validation loss does not decrease after 50 stratified epochs. We used a weight decay of 1e-06. The final global average pooling flattening the spatial dimension gives a descriptor with dimensionality 1024 transformed into D=128 by the LSTM. The final dimensionality considered in the loss is $2D$[2] and the two radii $r^{\{-,+\}} = \{0.042, 1.638\}$ have to be optimized together and cross-validated on a validation set. In high dimensional space, the volume of the hyper-sphere decreases when the feature descriptor dimension D increases [81]: thus, if D does change, the radii have to be changed accordingly. By increasing the dimensionality D of the final feature, the radii have to be increased as well to compensate for the diminished hyper-volume of the hyper-sphere. The cardinality F of the sequence of aligned frames as input to the recurrent model is 10. Since the sequential modeling is trained on sampled FF++ data, at

[2]The dimensionality is doubled since the results of the bi-directional streams are concatenated.

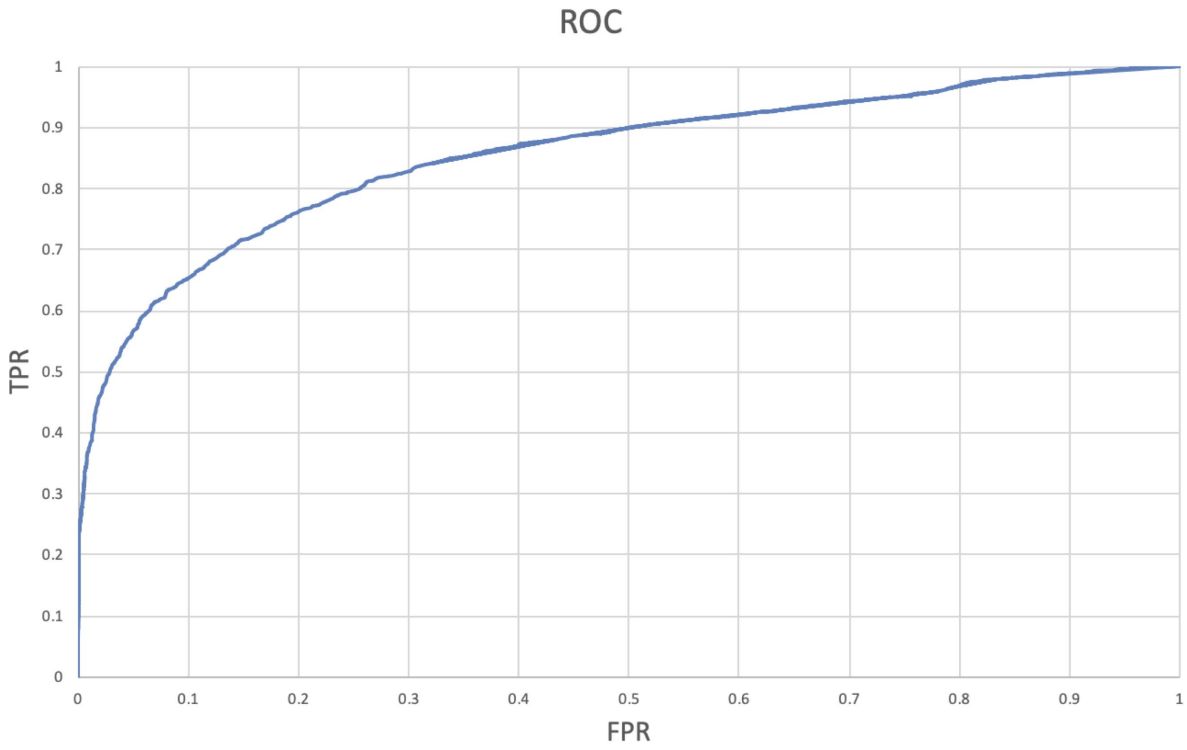


Figure 23: ROC curve for camera verification on the MFC19 Eval set.

inference time we take 1 frame over 7 to build the sequence. Faces are aligned with dlib [40]. If alignment fails, we revert back to [9]. In case of multiple detected faces, we select the largest detected face. Since FaceForensics++ has imbalanced labels (1:4), we oversample the natural faces twice and undersample randomly faces for each manipulation with a factor of two to get a proper balance, when training with multiple manipulations. We used average to perform video-level evaluation to aggregate all the scores within a video for all methods. When doing cross-testing, we use always the same model trained on FF++ on the four manipulations on high compression (c40).

FaceForensics++ (FF++): When training and evaluating on FF++, we follow the sampling strategy mentioned in [71] that selects 270 frames/video for the training and 110 frames/video for validation and testing. We evaluated both medium compression (c23) and high compression levels (c40) subsets. Table 4 shows a thorough comparison on FF++ [71] training and testing with four manipulations types (Deepfakes, FaceSwap, Face2Face, and NeuralTextures) along with the natural faces. Following [71], we trained a model for `c23` and another for `c40`. The table offers multiple evaluations metrics such as AUC, $pAUC_{10\%}$, $tAUC_{10\%}$ and $TAR_{10\%}$. In general, our approach has superior performance compared to Xception. In particular, we improved almost all frame-level performance for the medium compression case (c23), pushing the video-level AUC from 92% to 99%. The result is consistent for the other compression level but in general results are lower due to the low image quality; nevertheless our system improves video-level AUC from 86% to 91% along with other low false alarm video-level metrics. The table also reports the result of a self-supervised method Dual Spatial Pyramid for Exposing Face Warp Artifacts (DSP-FWA) [46]. Table 6a further shows the binary classification accuracies for several state-of-the-art face manipulation detection methods computed on FF++ [71]. Our approach scores the highest accuracies across manipulations for all the compression levels when trained on the four manipulations. It should be noted that a classifier exploiting the class imbalance here can get an accuracy of 80% by simply predicting all samples as fakes given that we have 140 real and 560 fake videos or similar balance at the frame level.

Celeb-DF: We evaluate how well our model transfers to Celeb-DF given that it is trained on FF++ with multiple manipulations. We do this with the goal of confirming that we optimized our method for better generalization across datasets, reaching a good balance between bias and variance. Table 5a shows a state-of-the-art evaluation at the frame- and video-level on the 518 test video of Celeb-DF, comparing it to other recent methods. Like other methods [71], we trained the model on FF++ to discern real faces

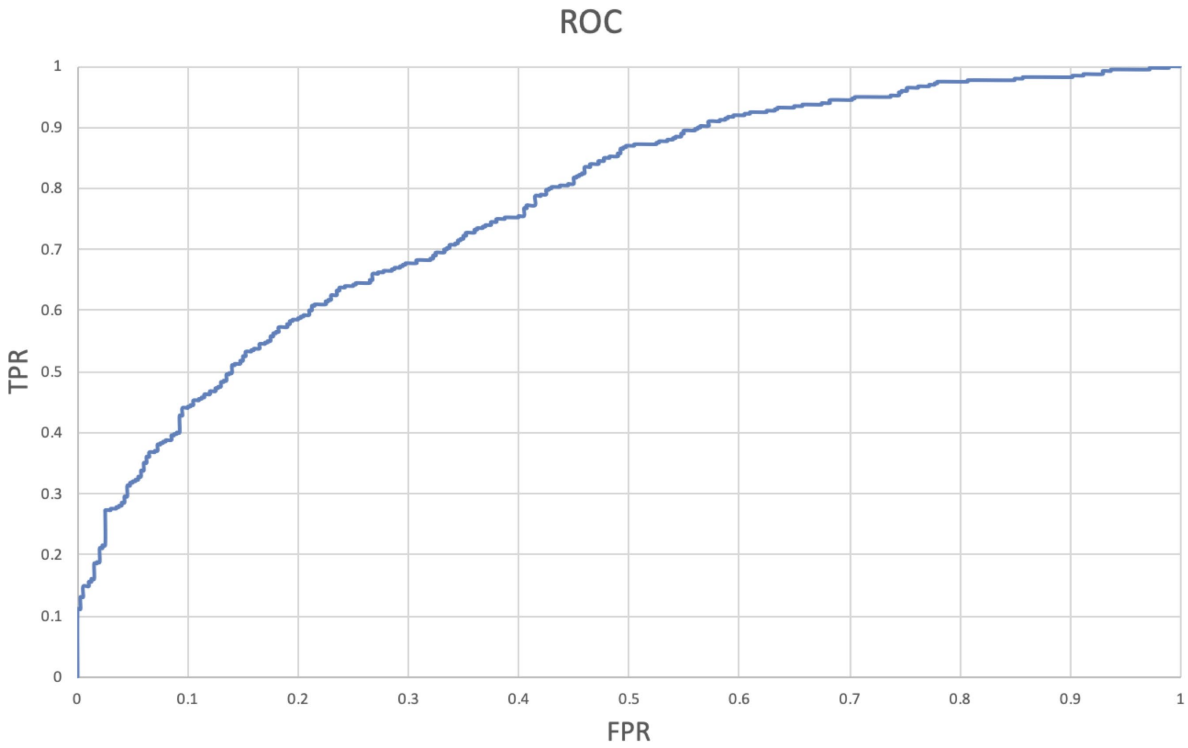


Figure 24: ROC curve for image pair verification on the MFC19 Eval set.

	HQ (c23)								LQ (c40)							
	Frame Level (~70K samples)			Video Level (700 samples)					Frame Level (~70K samples)			Video Level (700 samples)				
Methods	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$
DSP-FWA [46]	56.89	51.33	7.47	14.60	57.49	51.59	7.48	15.00	59.15	52.04	8.82	17.30	62.34	51.93	9.82	22.14
Xception [71]	92.30	87.71	**73.34**	81.21	92.50	89.20	58.21	82.85	83.93	**74.78**	**45.92**	**63.25**	86.75	**79.10**	39.06	68.75
Ours	**98.70**	**97.43**	65.29	**97.95**	**99.12**	**98.41**	**86.10**	**98.21**	**86.59**	69.71	40.41	62.48	**91.10**	76.57	**51.18**	**72.85**

Table 4: Frame-level and Video-level comparison on FF++. Multiple metrics reported for medium compression (c23) and high compression (c40) on FF++ comparing our method with XceptionNet [71] and DSP- FWA [46]. Results are reported on four manipulations.

versus four manipulation types at the c40 compression level. Table 5a reports a clear net improvement over the state-of-the-art, even when compared with recent methods that trained the model with self-supervision thereby, in theory, being less prone to overfitting, such as DSP-FWA [46]. Table 5b offers instead the classic evaluation performance in terms of AUC comparing our approach to the very recent method for digital face manipulation detection. We obtained higher AUC when compared to all the other methods on Celeb-DF while keeping an high AUC on FF++ on Deepfakes.

The Deepfake Detection Challenge (DFDC) Preview Dataset: We report video-level results on the "The Deepfake Detection Challenge (DFDC) preview set" using the evaluation described in [21]. This dataset contains approximately 5,250 videos of digitally manipulated and bona fide videos. As in [21], we used part of the training for cross validation for the two parameters available in our approach that are the optimal number of sequences and the distance $||\Phi(\mathbf{I}) - \mathbf{c}||_2$. We implemented five-fold cross-validation (20% of training retained for validation) and selected the best pair of parameters across the folds required to maximize the Log-Weighted Precision (log(wP)), with α=100, maintaining the desired level of recall. This procedure was repeated for different cutoff recalls ($R_{10\%}$, $R_{50\%}$, $R_{90\%}$). Although cross validation procedure aims to optimize the two parameters to keep a desired level of recall, meeting the same level of recall is not guaranteed when evaluating on the test set. This procedure simulates what can happen in real scenarios in which a system can be optimized on a validation set and then simply tested in the wild over millions of unlabeled data. For this reason, we report log(wP)@recall on the best validation fold under "valid" and the test set with "test-from-valid" using the parameters from validation. Alternatively, we also searched for the best log(wP) to exactly match the recall value on the test set and report those values under "test". Except for the above parameter selection, our method has not been

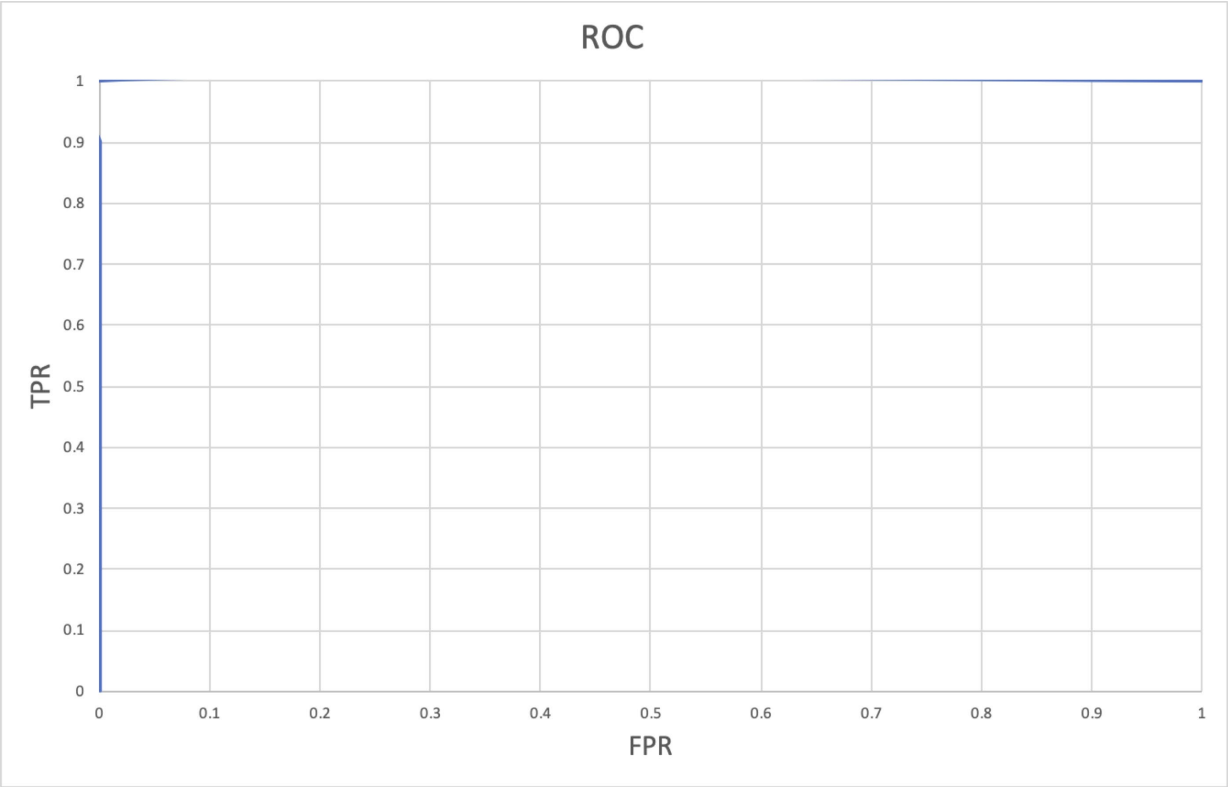


Figure 25: ROC curve for near-duplicate pair verification on the MFC19 Eval set.

re-trained on DFDC preview. Table 6b shows the evaluation results at the video level. Considering our results under "test," our method has slightly worse precision than XceptionNet [71] at $R_{10\%}$. However, if we optimize for high recall ($R_{90\%}$), we obtain a substantial boost in the log(wP), increasing log(wP) from -4.041 to -3.548. Moreover, we notice the following if we evaluate with the best hyper-parameters selected on the validation set our method maintains log(wP) better than other methods (-3.721) with a good recall of 0.943.

5.2 Physical Integrity

5.2.1 Robust Analysis of the Direction of Incident Light

The method is evaluated in two parts. We first show the performance of the estimator on a dataset with given segmentation. Then, we show the performance in a fully automated setting, with automatically segmented objects.

	Frame Level				Video Level			
Methods	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$	AUC	$pAUC_{10\%}$	$tAUC_{10\%}$	$TAR_{10\%}$
Xception-c40 [71]	65.86	54.49	12.23	22.97	69.70	57.18	16.85	34.70
DSP-FWA [46]	64.13	52.87	10.18	19.67	69.30	51.40	17.20	32.02
Xception-c23 [71]	66.65	53.05	10.21	19.83	73.04	52.77	9.45	18.82
Ours	**73.41**	**57.42**	**18.18**	**32.22**	**76.65**	**58.70**	**19.73**	**39.70**

(a)

Method	FF++ [71]	Celeb-DF [48]
Two-stream [31]	70.1	53.8
Meso4 [5]	84.7	54.8
MesoInception4	83.0	53.6
HeadPose [85]	47.3	54.6
FWA [46]	80.1	56.9
VA-MLP [54]	66.4	55.0
VA-LogReg	78.0	55.1
Xception-raw [71]	**99.7**	48.2
Xception-c23	**99.7**	65.3
Xception-c40	95.5	65.5
Multi-task [61]	76.3	54.3
Capsule [62]	96.6	57.5
DSP-FWA [46]	93.0	64.6
Ours	93.18	**73.41**

(b)

Table 5: Cross-dataset evaluation on Celeb-DF. (a) Frame- and video-level performance yet computed at a very low false alarm rate. Best competing methods on Celeb-DF are reported. Ours obtains a wide margin in all the low false alarm rate metrics (b) still performs well when tested on just deepfake class (93.18 %) AUC on FF++. Results for other methods are from [48].

Table 6: FF++ Accuracies and DFDC Preview Dataset. (a) Comparison of accuracies on FF++ (b) Video-level log(wP) for various recall rates.

Methods	HQ (c23)	LQ (c40)
[71] XceptionNet (Full Image)	74.78	70.32
[25] Steg. Features + SVM	70.97	55.98
[17] Cozzolino *et al.*	78.45	58.69
[7] Bayar and Stamm	82.97	66.84
[68] Rahmouni *et al.*	79.08	61.18
[5] MesoNet	83.10	70.47
[71] XceptionNet	95.73	81.00
Ours	**96.43**	**86.34**

(a)

Method	$R_{10\%}$	$R_{50\%}$	$R_{90\%}$
TamperNet [21]	-2.796@—	-3.864@—	-4.041@—
XceptionNet [71] (Face)	**-1.999@—**	**-3.012@—**	-4.081@—
XceptionNet [71] (Full)	-3.293@—	-3.835@—	-4.081@—
Ours (test)	-2.564@0.100	-3.152@0.501	**-3.548@0.901**
Ours (valid)	-2.311@0.090	-2.481@0.523	-2.678@0.918
Ours (test-from-valid)	-3.386@0.042	-3.433@0.440	-3.721@0.943

(b)

Table 7: AUC of ROC curves for binary classification into same or different lighting environments. The performance of the related methods drops significantly on the more challenging natural scenes. The proposed method performs consistently well on the natural scenes and outperforms the related methods.

	Laboratory (ALOI)			Natural Scenes (COCO)				
	All	Single	Multi	Person	Animal	Furniture	Vehicle	Mixed
Contour [38]	0.728	0.766	0.756	0.589	0.654	0.567	0.609	0.539
Contour ICE [69]	**0.740**	**0.776**	0.776	0.598	0.641	0.592	0.588	0.518
SIRFS [6]	0.738	0.763	**0.793**	0.580	0.665	0.581	0.559	0.524
Proposed	0.708	0.738	0.740	**0.716**	**0.735**	**0.633**	**0.657**	**0.677**

Classification of Lighting Environments on Pre-Segmented Objects We present results for laboratory data and for natural scenes. Laboratory data with single and multiple light sources are used from the ALOI dataset [26]. To further evaluate the performance in a realistic setting, we use the Common Objects in Context (COCO) dataset [14] to emulate splices. Here, we do not have ground truth illumination. Hence, make the common assumption that objects from different images exhibit different lighting environments [38]. Thus, the classification task is to determine whether two objects stem from the same image or from two different images.

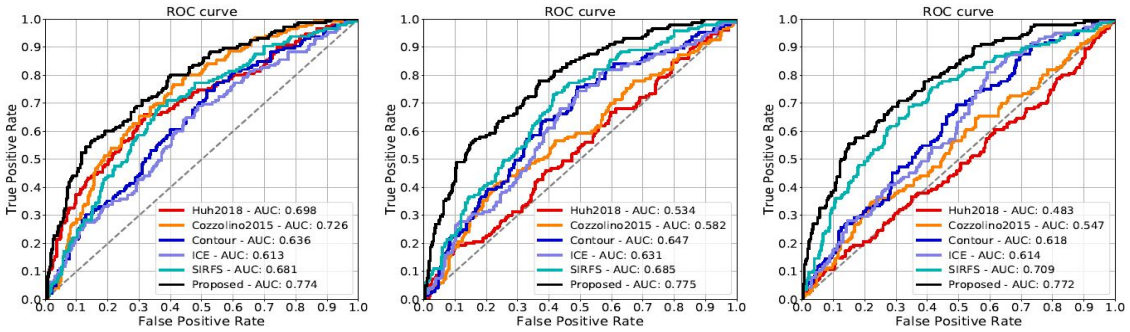
Table 7 shows the Area under the Curve (AUC) of the resulting Receiver Operating Characteristic (ROC) curves for binary classification into same or different lighting environments of pairs of images. The proposed method is compared to the contour lighting estimator by Johnson and Farid (Contour) [38], on the intrinsic-contour estimator by Riess *et al.* [69], and to the lighting environment estimator by Barron and Malick [6].

On the relatively clear laboratory data, the proposed method performs slightly worse than related analytic approaches. This changes on realistic objects from the COCO dataset. Here, for all object classes, the proposed method clearly outperforms the related works. This can be attributed to the robust inclusion of all object pixels, instead of working only on the contour, and a comparably simple inference process as opposed to SIRFS.

Fully Automated Splicing Detection To show the method performance in a fully automated setting, we create the OpenImages Splices (OIS) dataset. The source images of the dataset stem from the publicly available OpenImages V4 dataset [43], which contains about nine million images annotated with image-level labels and bounding-boxes. The OIS dataset consists of 450 images with two well visible persons each. In 150 of these images, one person is inserted from a different image. The pristine images are directly taken from the original URLs provided by the OpenImages dataset, and scaled to 1280 pixels in the larger image dimension. As the images in the dataset might themselves be preprocessed, we only consider the splicing of persons for manipulation detection.

The tampered images are created by selecting target and donor images using the provided image labels and Mask Region-based Convolutional Neural Network (R-CNN) for segmentation [32]. The target image is chosen to show exactly one well-visible person in foreground. The donor image is manually selected to find a person with reasonable semantic consistency to the target image. During creation of the dataset, no particular attention was paid to match illumination environments. As a consequence, the illumination can be accidentally consistent in spliced images, which we believe is a realistic situation for real splices. Care was

taken that both persons do not completely occlude each other upon splicing. The spliced



(a) OIS (b) OIS 960px, JPEG70 (c) OIS 600px, JPEG30

Figure 26: ROC curves for the fully-automated splicing classification on the three variants of the proposed OIS dataset.

persons are scaled and placed manually to fit the target image, and might be slightly blurred or copied with feathered edges, but no additional post-processing is applied. Target and donor image are scaled to 1280 pixels in the larger image dimension. The segmentation of Mask R-CNN is manually refined using GrabCut [72]. The quality of the splices is partly limited by the segmentation.

We believe that the OIS dataset presents an interesting benchmark for instance-level forensic methods, as the image provenance from the web (e.g., Flickr) is a plausible use case, and the detection of spliced persons is a semantically meaningful goal.

For evaluation, we additionally consider three variants of the dataset. The first variant uses the images as-is. The second variant downsamples the image such that the larger dimension is 960 pixels, and applies JPEG compression of quality 70. The third variant downsamples the image such that the larger dimension is only 600 pixels, and applies JPEG compression of quality 30. These cases mimick strong image degradations as they may be found, e.g., on forums or image boards in the internet.

The methods are compared to the same works as before (Contour [38], Contour ICE [69], and SIRFS [6], and additionally to two statistical methods by Huh *et al.* [37] and Cozzolino *et al.* [16].

The ROC curves for the three levels of image degradation are shown in Fig. 26. The setup is challenging for the statistical methods due to the strong image degradations, which remove most of the high-frequency information in the image. The physics-based methods are much less affected by these degradations. The proposed method clearly outperforms the related methods, with an almost constant AUC of about 0.77 across all three levels of degradation. More details can be found in the associated journal paper [55].

5.2.2 Segmentation-free Lighting Estimation

The segmentation-free lighting estimation is evaluated on the same dataset OIS and with the same experimental setup as the first experiment on the gradient-based estimator reported in Tab. 7. The segmentation-free approach sets the area outside of the segmented object to black, and scales the object bounding box to a square of 150 pixels. The results for this experiment are shown in Tab. 8. The proposed method outperforms the related works, and the very well-performing gradient-based estimator on all object classes except of "mixed" objects. This shows that this learning-based estimator is very well capable of estimating the lighting environment on pre-segmented objects.

Furthermore, this approach can also be used without any segmentation. We evaluate this on a synthetic dataset created from the multi-illuminant dataset by Murmann *et al.* [59]. The generated dataset contains 5000 images, whereof 2500 are spliced. The splices are generated by randomly changing a square region of an image with specific scene content and light setting to a different, randomly chosen, light setting. The images have a resolution of 750x500 pixels and the size of the spliced region is randomly sampled between 100 and 400 pixels per dimension. Qualitatively, the splices are differently hard to identify, depending on the size of the spliced region, the image content and the similarity of the randomly chosen light setting.

A heatmap is generated for splicing localization. To this end, each 150 × 150 pixels patch is compared to twelve randomly selected reference patches. These reference patches stem either from the same image,

Table 8: Comparison of the proposed segmentation-free method to the state-of-the-art for different object classes.

	Person	Animal	Furniture	Vehicle	Mixed
Contour [38]	0.589	0.654	0.567	0.609	0.539
Contour ICE [69]	0.589	0.641	0.592	0.588	0.518
SIRFS [6]	0.580	0.665	0.581	0.559	0.524
Gradient-based (previous Sec.)	0.716	0.735	0.633	0.657	**0.677**
Proposed	**0.753**	**0.749**	**0.651**	**0.663**	**0.641**

or the same image after downsampling or upsampling by a factor of 2, to enable a multi-scale analysis. The AUC of the ROC curve for this completely segmentation-free experiment is 0.718.

These experiments show that this learning-based approach to lighting estimation has the potential to fully replace all existing analytic approaches, while relaxing the requirement to compare pre-segmented object instances. Instead, it can operate on arbitrary image regions, similar to statistical forensic meth- ods. A publication on this method is currently in preparation.

5.2.3 Color Fingerprinting from the Scene and the Camera

The method is evaluated on several publicly available datasets, namely Columbia [35], DSO-1 [20], "Splices-In-The-Wild" dataset [37], and the Aligned Scenes dataset [30]. Additionally, we create a dataset that specifically aims at evaluating the detection of splices from different camera color pipelines, which we denote as "Spliced Color Pipeline".

The "Spliced Color Pipeline" dataset consists of 200 pristine images, consisting of randomly selected scenes from the developed RAW-to-final images in $\mathcal{S}^{\text{test}}$. It also consists of 200 manipulations, where one region of the scene is replaced by identical scene content, but developed with a randomly selected different camera pipeline. The region is a randomly selected superpixel with minimum size of $5 \cdot 10^4$ pixels (i.e., about three patch sizes), obtained with the segmentation algorithm by Felzenszwalb and Huttenlocher [24] with scale parameter 10 and $\sigma = 0.5$. Upon replacement, it is ensured that the average Lab-distance of the replaced and inserted regions is at least 5 to simulate local differences in within-camera color processing.

The images of all datasets are post-processed to simulate quality degradations of images distributed over the internet. The "Aligned Scenes" dataset applies downsampling and recompresses the images with JPEG qualities from 100 down to 10 in steps of 10 [30]. The remaining datasets are prepared in three variants: The high-quality (HQ) variant contains each image as-is. For the medium-quality (MQ) variant, each image is resized to a larger dimension of 1200 pixels, and JPEG compressed with quality 75. For the low-quality (LQ) variant, each image is resized to a larger dimension of 800 pixels, and JPEG compressed with quality 50.

Since the method primarily aims at manipulation localization, it is compared to the related localization methods "Noiseprint" (NP) [15], "Fighting Fake News" (FFN) [37], "Forensic Similarity" (FS) [57] and "Learned Color Representations" (LCR) [30]. The two aggregation variants of the proposed methods are denoted as "Medoid" and "Meanshift". The resulting ROC AUCs for manipulation localization are shown in Fig. 27 across the three considered quality levels (strength of postprocessing) for each dataset. Except for the DSO-1 dataset, the proposed method excells particularly in the analysis of low-quality images, since color is a low-frequency property that is relatively robust to strong JPEG compression and downsampling.

A similar trend can be observed for splicing detection. Here, we selected for each method the best threshold on the generated heatmap to separate pristine and manipulated images. Note that the "In the Wild" dataset is omitted here, since it only contains manipulations. The resulting accuracies again show that the proposed method is particularly stable under strong image degradations, where it outperforms all competing methods.

5.2.4 Fingerprinting of JPEG Library Chroma Subsampling

The properties of the chroma subsampling artifact are evaluated on the 1491 images from the Dresden database [27]. We convert the RAW images to JPEG using `dcraw` and `cjpeg` from `libjpeg v9a`. Com-

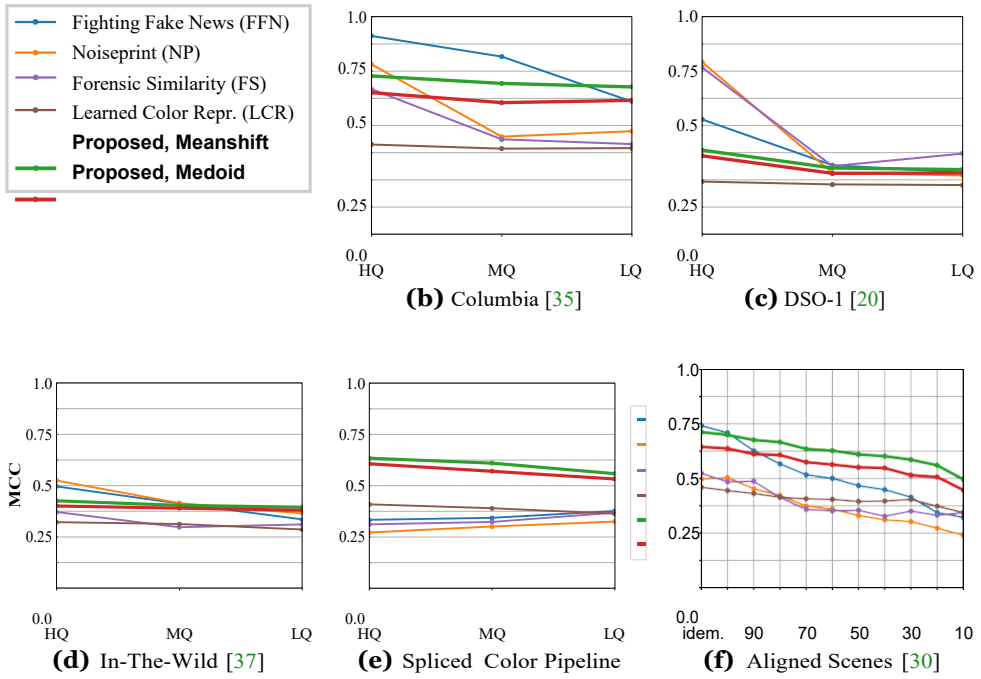


Figure 27: Matthews Correlation Coefficient for color-based splicing localization on various datasets.

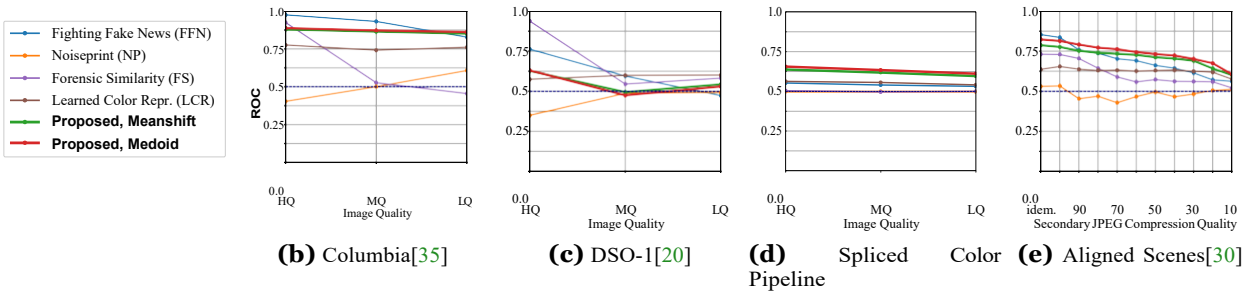


Figure 28: ROC AUC for color-based splicing detection on various datasets.

mand line switches allow to generate images both with *Discrete Cosine Transform (DCT) scaling* and with *simple scaling*, where the latter includes the artifact. In all experiments, care was taken that the type of subsampling is the only difference in the processing of the images. All experiments are performed with 4:2:0 chroma subsampling, but the same artifact is analogously part of 4:2:2 chroma subsampling. We distinguish both scaling variants with a Support Vector Machine (SVM), where one SVM is trained for each quality factor. The training is performed separately for each quality factor on 90% of the images using 10-fold cross-validation. Testing is performed on the remaining images.

Figure 29 (left) shows the accuracy per quality factor. Accuracies range around 98% for JPEG quality 90 and better. With decreasing JPEG quality, the classifier's effectiveness drops, which is expected due to the suppression of high-frequencies.

Figure 29 (right) shows how well the artifact can be recovered when four common post processing operations are applied, namely recompression, gamma adjustment, additive noise, and image scaling. To this end, the RAW images from the Dresden database are first converted to JPEG with *simple scaling* and quality 100. We then apply *DCT upsampling*, perform the post-processing operation in image space, and recompress the resulting image with *DCT scaling* and quality factor 100. The results show that recompressing the image again at a lower quality level impacts detectability, with high accuracies only for JPEG quality 98 or higher. The artifact is quite robust to gamma correction, but susceptible also to additive Gaussian noise and resampling. Hence, the artifact can only be recovered for high-quality images, and as such is likely most useful for fingerprinting of images that are supposed to stem directly from a camera, or where the background of an image carries the artifact, but an inserted or edited part of the image does not. More details can be found in our paper [50].

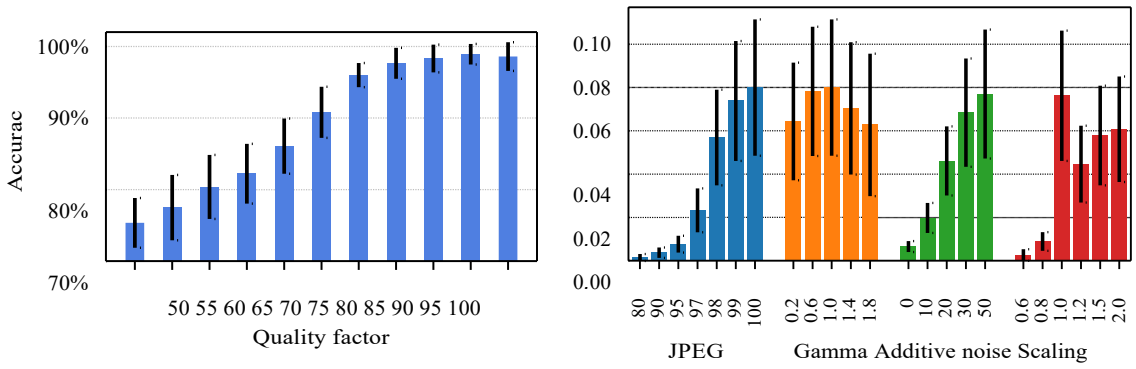


Figure 29: Left: Accuracy to distinguish *simple scaling* and *DCT scaling* with a linear SVM on block correlations. The detection accuracy decreases with lower JPEG quality factors. Right: Correlation scores in Cb channel after applying one of four common post-processing operations: JPEG compression (quality factor), gamma correction (gamma), corruption by additive noise (Signal-to-Noise Ratio (SNR) in dB), scaling (scale factor).

Table 9: Results for algorithm fingerprinting and patch discrimination. Evaluation on generated data with different network variants.

Architecture	Fingerprinting Accuracy	Discrimination AUC
Xception [13]	**0.97**	**0.997**
ResNet-50 [33]	0.91	0.989
MISLNet [8]	0.90	0.983
MesoNet [5]	0.62	0.958

5.2.5 Fingerprinting of Depth Image Calculation in Cameras

Depth map fingerprinting is first evaluated on the 7481 images from the Kitti dataset are split into 5404, 954, and 1123 images for training, validation, and test. On the test images, we classify the center patch per class. The average classification accuracy is shown in the middle column of Tab. 9. Xception performs best with a remarkable accuracy of 0.97. ResNet-50 and MISLNet perform slightly worse with accuracies of 0.91 and 0.90. MesoNet only achieves an accuracy of 0.62.

For patch discrimination, the trained network is integrated into a Siamese architecture. We only report results for retraining the new top layers, as additional end-to-end training did not further improve results. For evaluation, 13464 patch pairs are randomly chosen, where the patches in 50% of the pairs stem from the same device, and in 50% of the pairs from different devices. Area under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve are reported in the right column of Tab. 9. Consistent with the fingerprinting results, Xception performs best with an excellent AUC of 0.997. Overall, all networks perform very well with AUC values between 0.958 and 0.989.

We further use the collected smartphone data for patch discrimination. First, the models are evaluated without further training to test the generalization of the features learned from generated data. Then,we perform few-shot fine-tuning of the pretrained networks, by retraining the models with five images from a single scene of the dataset. We analyze the performance by evaluating samples for specific pairs of devices. The resulting ROC curves are shown in Fig. 30. Overall, the performance is quite encouraging. We note that the model cannot reliably discriminate patches from smartphones of the same manufacturers, which is expected due to the assumed similarities in hardware and software between related models. Overall, the experiments show that the generalization of the learning from generated data is sufficient for few-shot training of powerful CNN architectures with little data. More details can be found in our paper [56].

5.3 Semantic Integrity

5.3.1 Provenance Filtering

Data corpora The method was evaluated using a number of NIST NIMBLE dataset releases. The NC2016 Web dataset has 724 probes and a world set of 1124 reference images. Out of 724 probes, we found 128 that have spliced objects corresponding to 264 donor images in the world set of 1124 reference

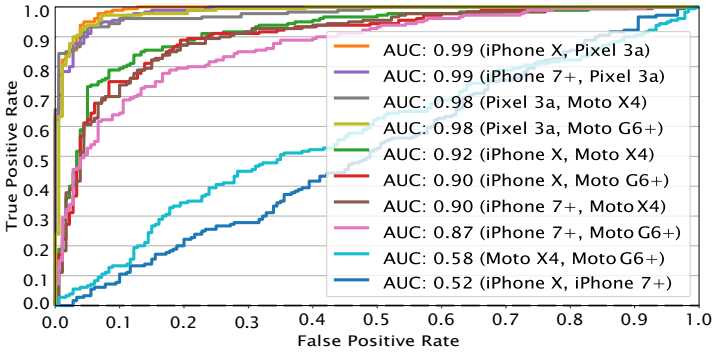


Figure 30: ROC curves for evaluating samples of specific pairs of devices with the Xception variant.

Table 10: Impact of the segmentation scheme on donor detection (NC2016).

Method	Recall % @Top 10
Fisher w/o segmentation	13.6
DML (CARS) w/o segmentation	18.9
DML (CARS) w/ semantic segmentation	31.4

images. The NC2017 Dev1Beta4 dataset has 65 probe images and a world set of 1631 reference images. The NC2017Dev3Beta1 dataset has 2260 probe images and a world set of 3441 reference images. The NIMBLE Evaluation world set has 2992 probe images and 1008681 reference images. The ground truth is given for 993 out of 2992 probe images.

Segmentation vs. Whole-image Matching for Donor Detection We ran the first set of experiments on NC2016 Web to evaluate several donor detection strategies. We ran an image-to-image matching using 512D Fisher vectors embedding raw Overfeat [84] features and got 100% recall rate from top-1 base detection results. We also used a Deep Metric Learning (DML) model [64] to represent the image using a 128D feature vector. We experimented all three pre-trained models for DML coming from different training sets (CARS, online products, CUB) mentioned in that paper. The manipulated regions for each probe were obtained by subtracting the probe from its base, and were used as queries to search for donors. We tried to use both the homogeneous regions returned by the Berkeley Semantic Segmentation algorithm [49] and the whole image can search candidates for donor detection. We took the maximum of cosine similarities between a query and all candidates of an image as the final similarity score. The recall rates of donors from top 10 results of these experiments are shown in Table 10 . By comparing the first two rows of results, the DML features from the CARS model outperforms the Fisher vector method. By comparing the 2nd and the 3rd rows of results, semantically segmented candidates have significantly better performance than the whole image.

Comparison of Features We made two attempts to further improve our core algorithm for donor detection. First, keeping Berkerly's semantic segmentation unchanged, we tried the pre-trained DML models for the online_products and CUB datasets. We also used the Inception V3 [78] and VGG-16 [76] pre-trained models. The recall rates of donor detection from top 10 results are shown in Table 11 . The best performance is obtained from using the VGG-16 model.

The second attempt was to replace semantic segmentation with evenly sampled rectangular sliding windows. We used 188 sub-images from the 6 scales of sliding windows and extracted DML (CUB) features from each sub-image. Table 12 indicates significant improvement of donor detection is obtained from using the sliding windows.

Results on Evaluation Datasets We compared with two other systems on the NC2017 dataset. Our system on NC2017 used the experiment conditions that have been evaluated as having the best performance on the dev set, i.e., 6-scale sliding window segmentation and VGG-16 features. Locality-sensitive hashing is deployed to speed up the computation of similarity scores in all stages described in Section II. The system returns a number of results, each corresponding to a certain operation from the

Table 11: Comparison of features using Semantic Segmentation on NC2016.

Method	Recall % @Top 10
DML (CARS)	31.4
DML (online_products)	45.1
DML (CUB)	61.7
Inception V3	81.1
VGG-16	81.8

Table 12: Comparison of Semantic Segmentation vs. sliding-window sampled sub-images using DML (CUB) features on NC2016.

Method	Recall % @Top 10
Semantic segmentation	61.7
6-scale sliding window	81.8

following:

- Base detection
- Donor detection
- Splice detection in ref images + hue rescoring
- Probe-as-donor detection + multi-dimensional (MD) brute-force matching + hue value-based rescoring
- Indirect provenance detection

A certain number of top candidates from each run of search are collected and aggregated to create up to 300 candidates for each probe. Fig. 9 illustrates how these results (on the NC2017 Evaluation set) that have somewhat not very high recall produce a much better result when merged. Table 13 shows our system performance on MFC20 Eval set.

6 Conclusions

6.1 Noiseprint

We proposed a deep learning method to extract a noise residual, called noiseprint, where the scene content is largely suppressed and model-related artifacts are enhanced. Therefore, a noiseprint bears traces of the ideal camera model fingerprint much like a PRNU residual bears traces of the ideal device fingerprint. In noiseprints, however, the signal of interest is much stronger than in PRNU residuals, allowing for the reliable accomplishment of many forensic tasks. Experiments show that noiseprint can be a promising tool for the image forgery localization task for different type of manipulations.

6.2 An end-to-end trainable approach for image forgery detection

We proposed a new CNN-based framework for image forgery detection. Thanks to suitable architectural solutions, it allows one to process jointly information gathered at full-resolution from the whole image. Moreover, the framework can be trained end-to-end based only on weak (image-level) supervision. We proved the effectiveness of this solution by extensive performance analysis on forensic datasets widespread

Table 13: Provenance filtering performance in the MFC20 Evaluation

Recall % @Top 50	Recall % @Top 100	Recall % @Top 200	Recall % @Top 300
0.814	0.846	0.867	0.877

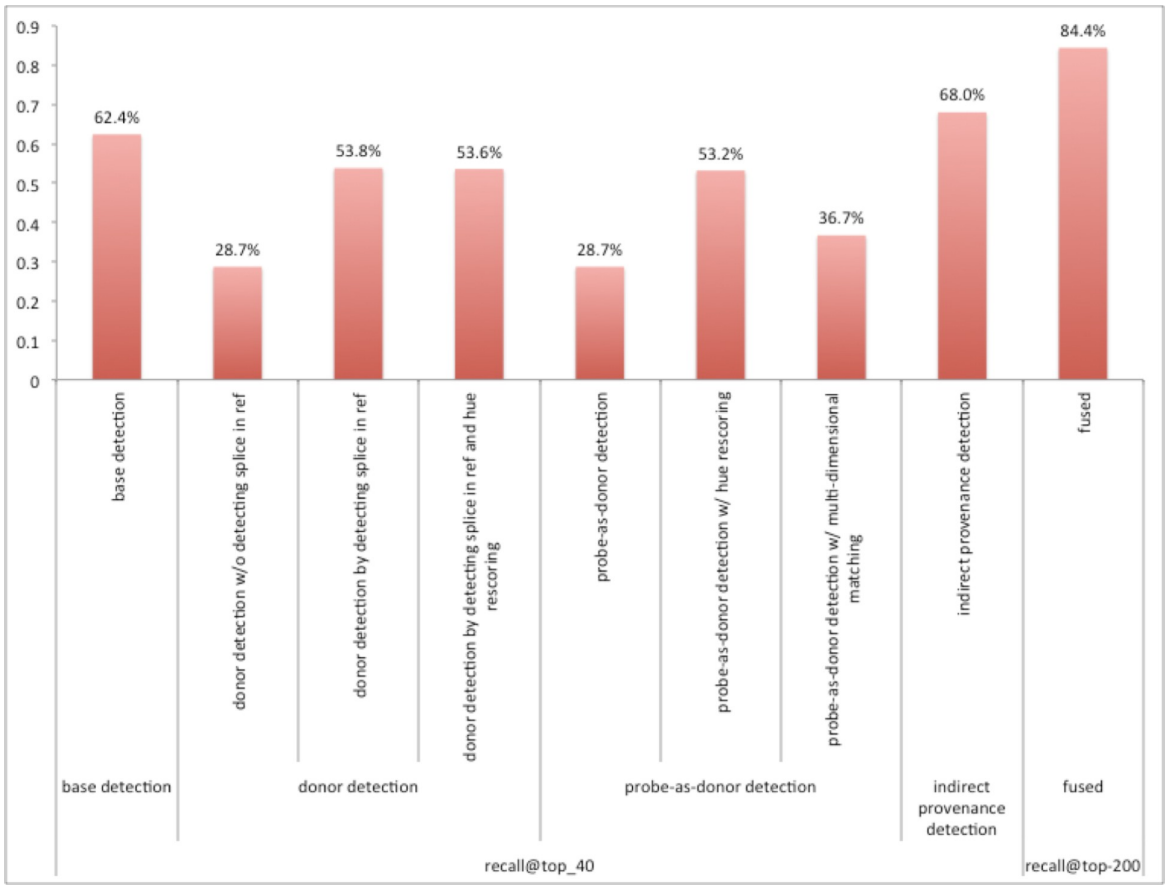


Figure 31: Recall rates measured for the provenance filtering results from single operations and the merged result.

in the community. A large performance gain is observed in all cases with respect to all reference methods. In addition, the framework can be also recast to provide localization information, both in supervised and unsupervised modality.

6.3 GAN fingerprints

The goal of this work was to prove the existence of GAN fingerprints and their value for reliable forensic analyses. We have demonstrated that each GAN leaves its specific fingerprint in the images it generates, just like real-world cameras mark acquired images with traces of their photo-response non-uniformity pattern. Source identification experiments with several popular GANs show such fingerprints to represent a precious asset for forensic analyses.

6.4 Video facial manipulation detection

While current state-of-the-art facial image manipulation methods exhibit visually stunning results, we demonstrated that they can be detected by trained forgery detectors. To this end we introduced a novel dataset of videos of manipulated faces that includes four different type of facial manipulations, It is particularly encouraging that also the challenging case of low-quality video can be tackled by learning-based approaches, where humans and hand-crafted features exhibit difficulties.

6.5 Video Copy-move Detection and Localization

We proposed a method for the detection and localization of video copy-moves. Since keypoint-based approaches are ineffective with most occlusive forgeries, we focused on dense-field methods. With this approach, the main issue is complexity, especially for videos, cursed by their huge data size. To deal with this problem we resorted to a fast randomized patch matching algorithm, a hierarchical analysis

strategy, and parallel implementation. Experiments confirm that the proposed method has an excellent detection and localization ability, also for occlusive copy-moves, and even in adverse scenarios including rotated copy-moves and compressed videos. Moreover, the running time is much reduced w.r.t. linear search, enabling practical video analysis.

6.6 Image Manipulation

We have developed a novel end-to-end DNN solution to image forgery localization called ManTra-Net. Our extensive experimental results using only pre-trained models demonstrate that the proposed ManTra- Net is sensitive to subtle manipulations, and robust to post processing disguising manipulations, and that it attains good generalizability to unseen data and unknown manipulation types

6.7 Camera Identification

We developed a camera identification model with good performance for camera verification and image pair verification. The camera identification model has very reliable performance for near-duplicate verification for the MFC2019 dataset. This is potentially useful for improving the accuracy in the near-duplicate clustering stage for provenance filtering.

6.8 Two-branch Recurrent Network for Isolating Deepfakes in Videos

We developed a method for video-based deepfake detection that uses a recurrent model to process sequences of aligned faces using a two-branch backbone to fuse information across the color and frequency domain. The method is supervised with a novel loss function that isolates manipulated face sequences in the feature space. We have shown results on FaceForensics++, Celeb-DF, and DFDC that outperform or are on par with state-of-the-art. In the long term, we plan to augment our model with an explainability mechanism that does not need any pixel-wise supervision for face manipulations.

6.9 Robust Analysis of the Direction of Incident Light

We proposed a new method for estimating the lighting environment on an object in the image plane. In contrast to previous works, the method operates on all pixels of the object, which dramatically increases the robustness towards common failure cases of lighting-based methods, such as slight mis-segmentations, partial self-shadowing or occlusions. It also makes the method remarkably robust to common post processing operations such as strong JPEG compression and significant down-sampling.

6.10 Segmentation-free Lighting Estimation

We proposed a learning-based method for estimating the lighting environment on arbitrary image patches. This completely removes the requirement of pre-segmented objects, and hence enables application of the method on a considerably broader range of scenes. Although the method is learning-based, it also exhibits excellent robustness to common post processing operations such as JPEG recompression and down-sampling.

6.11 Color Fingerprinting from the Scene and the Camera

We proposed a method to learn a metric space for fingerprinting the color formation in cameras. The proposed method maps variations in illumination and in camera-internal color processing far apart, which is subsequently used for splicing localization. The method particularly excells in its robustness to common postprocessing operations such as JPEG compression and downsampling.

6.12 Fingerprinting of JPEG Library Chroma Subsampling

We present a characteristic artifact in the chroma subsampling of popular implementations of the JPEG library. The artifact can be detected in high-quality images of JPEG quality 90 and beyond. It has applications in fingerprinting the origin of images, and also in exposing local manipulations that lead distort the artifact.

6.13 Fingerprinting of Depth Image Calculation in Cameras

We present a new forensic cue for modern smartphones, namely to exploit JPEG-embedded depth information for forensic analysis. We show that the depth image exhibits characteristic traces for the hardware setup and the computational algorithm to compute the depth image. This subtle cue imposes the additional constraint on a manipulator to also consistently edit the depth image to create a believable forgery.

6.14 Provenance Filtering

We developed an end-to-end provenance filtering system with excellent performance and all the feature extraction, indexing and retrieval modules implemented as specified by the program-defined API. The evaluation results showed our system is capable of indexing million-image datasets for searching.

References

[1] CNN - business - when seeing is no longer believing inside the pentagon's race against deepfake videos. https://www.cnn.com/interactive/2019/01/business/pentagons-race-against-deepfakes/. 15

[2] DeepFaceLab. https://github.com/iperov/DeepFaceLab. 15

[3] ZAO app. https://apps.apple.com/cn/app/zao/. 15

[4] Deepfake videos pose a threat, but dumbfakes may be worse. https://apnews.com/e810e38894bf4686ad9d0839b6cef93d, 2019. 5, 15

[5] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In *WIFS*, pages 1–7. IEEE, 2018. 31, 32, 36

[6] Jonathan T. Barron and Jitendra Malik. Shape, illumination, and Reflectance from Shading. *TPAMI*, 37(8):1670–1687, August 2014. 32, 33, 34

[7] Belhassen Bayar and Matthew C Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *ACM Workshop on Information Hiding and Multimedia Security*, pages 5–10, 2016. 32

[8] Belhassen Bayar and Matthew C Stamm. Constrained Convolutional Neural Networks: A New Approach Towards General Purpose Image Manipulation Detection. *TIFS*, 13(11):2691–2706, November 2018. 36

[9] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017. 29

[10] Peter Burt and Edward Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on communications*, 31(4):532–540, 1983. 15

[11] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. Learning Photographic Global Tonal Adjustment with a Database of Input / Output Image Pairs. In *CVPR*, pages 97–104, 2011. 19

[12] Yizong Cheng. Mean Shift, Mode Seeking, and Clustering. *TPAMI*, 17(8):790–799, August 1995. 19

[13] Francois Chollet. Xception: Deep Learning With Depthwise Separable Convolutions. In *CVPR*, pages 1251–1258, 2017. 36

[14] COCO Consortium. COCO - Common Object in Context. 32

[15] D. Cozzolino and L. Verdoliva. Noiseprint: A CNN-based camera model fingerprint. *IEEE Transactions on Information Forensics and Security*, 15(1):14–27, 2020. 9, 34

[16] Davide Cozzolino, Gianni Poggi, and Luisa Verdoliva. Splicebuster: A New Blind Image Splicing Detector. In *WIFS*, pages 1–6, November 2015. 33

[17] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *ACM Workshop on Information Hiding and Multimedia Security*, pages 159–164, 2017. 32

[18] L. D'Amiano, D. Cozzolino, G. Poggi, and L. Verdoliva. A PatchMatch-based dense-field algorithm for video copy-move detection and localization. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(3):669–682, March 2019. 13

[19] Duc-Tien Dang-Nguyen, Cecilia Pasquini, Valentina Conotter, and Giulia Boato. Raise: a raw images dataset for digital image forensics. In *ACM Multimedia Systems Conference*, pages 219–224, 2015. 19

[20] Tiago Jose de Carvalho, Christian Riess, Elli Angelopoulou, Hélio Pedrini, and Anderson de Rezende Rocha. Exposing digital image forgeries by illumination color classification. *IEEE Trans. Information Forensics and Security*, 8(7):1182–1194, 2013. 34, 35

[21] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The Deep- fake Detection Challenge (DFDC) Preview Dataset. *arXiv:1910.08854 [cs]*, October 2019. arXiv: 1910.08854. 15, 27, 30, 32

[22] Pedro M Domingos. A few useful things to know about machine learning. *Commun. acm*, 55(10):78–87, 2012. 15

[23] W. Fan, K. Wang, F. Cayre, and Z. Xiong. 3D Lighting-based Image Forgery Detection using Shape-from-Shading. In *European Signal Processing Conference (EUSIPCO)*, pages 1777–1781, August 2012. 17

[24] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004. 34

[25] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *TIFS*, 7(3):868–882, 2012. 32

[26] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. The Amsterdam Library of Object Images. *IJCV*, 61(1):103–112, 2005. 32

[27] Thomas Gloe and Rainer Böhme. The 'Dresden Image Database' for Benchmarking Digital Image Forensics. In *ACM Symposium on Applied Computing*, pages 1584–1590, 2010. 34

[28] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 15

[29] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *AVSS*, pages 1–6. IEEE, 2018. 15

[30] Benjamin Hadwiger, Daniele Baracchi, Alessandro Piva, and Christian Riess. Towards Learned Color Representations for Image Splicing Detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8281–8285, 2019. 34, 35

[31] Xintong Han, Vlad Morariu, Peng IS Larry Davis, et al. Two-stream neural networks for tampered face detection. In *CVPR Workshops*, pages 19–27, 2017. 31

[32] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, October 2017. 32

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 36

[34] Silvan Heller, Luca Rossetto, and Heiko Schuldt. The PS-Battles Dataset – an Image Collection for Image Manipulation Detection. *CoRR*, abs/1804.04866, 2018. 15

[35] Yu-Feng Hsu and Shih-Fu Chang. Detecting image splicing using geometry invariants and camera characteristics consistency. In *Proceedings of the 2006 IEEE International Conference on Multimedia and Expo, ICME 2006, July 9-12 2006, Toronto, Ontario, Canada*, pages 549–552. IEEE Computer Society, 2006. 34, 35

[36] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 15

[37] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A. Efros. Fighting Fake News: Image Splice Detection via Learned Self-Consistency. In *ECCV*, pages 106–124, 2018. 33, 34, 35

[38] Micah K. Johnson and Hany Farid. Exposing Digital Forgeries in Complex Lighting Environments. *TIFS*, 2(3):450–461, September 2007. 17, 32, 33, 34

[39] Eric Kee and Hany Farid. Exposing Digital Forgeries from 3-D Lighting Environments. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, December 2010. 17

[40] Davis E King. Dlib-ml: A machine learning toolkit. *JMLR*, 10:1755–1758, 2009. 29

[41] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 15

[42] Pavel Korshunov and Sébastien Marcel. Deepfakes: a New Threat to Face Recognition? Assessment and Detection. *arXiv preprint arXiv:1812.08685*, 2018. 27

[43] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Shahab Kamali, Matteo Malloci, Jordi Pont- Tuset, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. OpenImages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017. 32

[44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 15

[45] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 15

[46] Yuezun Li and Siwei Lyu. Exposing deepfake videos by detecting face warping artifacts. In *CVPR Workshops*, June 2019. 7, 29, 30, 31

[47] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A new dataset for deepfake forensics. *arXiv preprint arXiv:1909.12962*, 2019. 15

[48] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df (v2): A new dataset for deepfake forensics. In *CVPR*, 2020. 7, 27, 31

[49] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 37

[50] Benedikt Lorch and Christian Riess. Image Forensics from Chroma Subsampling of High-Quality JPEG Images. In *ACM Workshop on Information Hiding and Security (IH&MMSec)*, pages 101– 106, 2019. 35

[51] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 5, 16

[52] F. Marra, D. Gragnaniello, L. Verdoliva, and G. Poggi. Do GANs leave artificial fingerprints? In *2nd IEEE International Workshop on Fake MultiMedia*, March 2019. 11

[53] F. Marra, D. Gragnaniello, L. Verdoliva, and G. Poggi. A Full-Image Full-Resolution End-to-End-Trainable CNN Framework for Image Forgery Detection. *IEEE Access*, 8:133488–133502, 2020. 10

[54] Falko Matern, Christian Riess, and Marc Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *WACV Workshops*, pages 83–92. IEEE, 2019. 31

[55] Falko Matern, Christian Riess, and Marc Stamminger. Gradient-Based Illumination Description for Image Forgery Detection. *TIFS*, 15(1):1303–1317, December 2019. 17, 33

[56] Falko Matern, Christian Riess, and Marc Stamminger. Depth Map Fingerprinting and Splicing Detection. In *ICASSSP*, pages 2782–2786, 2020. 36

[57] Owen Mayer and Matthew C. Stamm. Forensic Similarity for Digital Images. *TIFS*, 15:1331–1346, 2020. 34

[58] Donna Katzman McClish. Analyzing a portion of the roc curve. *Medical Decision Making*, 9(3):190–195, 1989. 28

[59] Lukas Murmann, Michael Gharbi, Miika Aittala, and Fredo Durand. A Dataset of Multi-Illumination Images in the Wild. In *ICCV*, pages 4080–4089, 2019. 18, 33

[60] Seonghyeon Nam and Seon Joo Kim. Modelling the Scene Dependent Imaging in Cameras with a Deep Neural Network. In *ICCV*, pages 1726–1734, 2017. 19

[61] Huy H Nguyen, Fuming Fang, Junichi Yamagishi, and Isao Echizen. Multi-task learning for detecting and segmenting manipulated facial images and videos. In *BTAS*, 2019. 31

[62] Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. In *ICASSSP*, pages 2307–2311. IEEE, 2019. 31

[63] Yuval Nirkin, Iacopo Masi, Anh Tran, Tal Hassner, and Gerard Medioni. On face segmentation, face swapping, and face perception. In *AFGR*, 2018. 15

[64] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016. 37

[65] Domingos Pedro. A unified bias-variance decomposition and its applications. In *17th International Conference on Machine Learning*, pages 231–238, 2000. 15

[66] Bo Peng, Wei Wang, Jing Dong, and Tieniu Tan. Automatic Detection of 3D Lighting Inconsisten- cies via a Facial Landmark based Morphable Model. In *IEEE International Conference on Image Processing (ICIP)*, pages 3932–3936, September 2016. 17

[67] Alex P. Pentland. Finding the illuminant direction. *Journal of the Optical Society of America*, 72(4):448–455, April 1982. 17

[68] Nicolas Rahmouni, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Distinguishing computer graphics from natural images using convolution neural networks. In *WIFS*, pages 1–6, 2017. 32

[69] Christian Riess, Mathias Unberath, Farzad Naderi, Sven Pfaller, Marc Stamminger, and Elli An- gelopoulou. Handling Multiple Materials for Exposure of Digital Forgeries using 2-D Lighting En- vironments. *Multimedia Tools and Applications*, 76(4):4747–4764, February 2016. 32, 33, 34

[70] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. Faceforensics++: Learning to detect manipulated facial images. In *International Conference on Computer Vision*, 2019. 12

[71] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *ICCV*, 2019. 7, 15, 17, 27, 29, 30, 31, 32

[72] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts. In *SIGGRAPH*, pages 309–314, 2004. 33

[73] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, pages 4393–4402, 2018. 16, 17

[74] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natara- jan. Recurrent convolutional strategies for face manipulation detection in videos. In *CVPR Work- shops*, pages 80–87, 2019. 15

[75] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 17

[76] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR, abs/1409.1556*, 2014. 37

[77] Joel Stehouwer, Hao Dang, Feng Liu, Xiaoming Liu, and Anil Jain. On the detection of digital face manipulation. In *CVPR*, 2020. 27

[78] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 37

[79] The New York Times. Distorted videos of nancy pelosi spread on facebook and twitter, helped by trump. https://www.nytimes.com/2019/05/24/us/politics/pelosi-doctored-video.html, 2019. 5, 15

[80] Giorgio Valentini and Thomas G Dietterich. Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. *JMLR*, 5(Jul):725–775, 2004. 15

[81] Eric W Weisstein. Hypersphere. 2002. 28

[82] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 17

[83] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, Oct 2017. 17

[84] David Eigen Xiang Zhang Michaël Mathieu Rob Fergus Yann LeCun Pierre Sermanet. Over-feat: Integrated recognition, localization and detection using convolutional networks. *CoRR, page abs/1312.6229*, 2013. 37

[85] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In *ICASSSP*, pages 8261–8265. IEEE, 2019. 31

[86] Premkumar Natarajan Yue Wu, Wael AbdAlmageed. Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. *CVPR 2019: 9543-9552*, 2019. 14, 26

Acronyms

AE AutoEncoder. 15

AI Artificial Intelligence. 15

AUC Area under the Curve. 6, 7, 27, 32–36

Celeb-DF Celebrity DeepFake Dataset. 27

CNN Convolutional Neural Network. 5, 9, 10, 14, 18, 20, 21, 25, 36

COCO Common Objects in Context. 32

DCNN Deep Convolutional Neural Network. 15

DCT Discrete Cosine Transform. 6, 35, 36

DFDC Deefake Detection Challenge Preview Dataset. 27

DSP-FWA Dual Spatial Pyramid for Exposing Face Warp Artifacts. 29, 30

FAR False Acceptance Rate. 27

FF++ FaceForensics++. 27

GAN Generative Adversarial Network. 3, 5, 10, 11, 15, 24–26

JPEG Joint Photographic Experts Group. 3, 6, 20, 33–36

LoG Laplacian of Gaussian. 5, 16

log(wP) Log-Weighted Precision. 30

LSTM Long Short-Term Memory. 5, 15, 16, 28

MesoNet Mesoscopic Analysis Neural Network. 20, 36

MISLNet Multimedia and Information Security Lab Neural Network. 20, 36

OIS OpenImages Splices. 6, 32, 33

pAUC Partial Area Under the Curve. 28

PRNU Photo-Response Non-Uniformity. 9, 11

R-CNN Region-based Convolutional Neural Network. 32, 33

ResNet Residual Neural Network. 19, 20, 36

ROC Receiver Operating Characteristic. 6, 7, 27, 32–37

SNR Signal-to-Noise Ratio. 6, 36

SVDD Support Vector Data Description. 16

SVM Support Vector Machine. 6, 35, 36

t-SNE t Distributed Stochastic Neighbor Embedding. 5, 16, 17

TAR True Acceptance Rate. 27

tAUC Truncated AUC. 28

Xception Extreme Inception Neural Network. 6, 20, 36, 37