Published in IEEE Transactions on Information Forensics and Security.

Deep Metric Color Embeddings for Splicing Localization in Severely Degraded Images

Benjamin Hadwiger, Christian Riess, *Senior Member, IEEE*

***Abstract*—One common task in image forensics is to detect spliced images, where multiple source images are composed to one output image. Most of the currently best performing splicing detectors leverage high-frequency artifacts. However, after an image underwent strong compression, most of the high frequency artifacts are not available anymore.**

In this work, we explore an alternative approach to splicing detection, which is potentially better suited for images in-the-wild, subject to strong compression and downsampling. Our proposal is to model the color formation of an image. The color formation largely depends on variations at the scale of scene objects, and is hence much less dependent on high-frequency artifacts. We learn a deep metric space that is on one hand sensitive to illumination color and camera white-point estimation, but on the other hand insensitive to variations in object color. Large distances in the embedding space indicate that two image regions either stem from different scenes or different cameras. In our evaluation, we show that the proposed embedding space outperforms the state of the art on images that have been subject to strong compression and downsampling. We confirm in two further experiments the dual nature of the metric space, namely to both characterize the acquisition camera and the scene illuminant color. As such, this work resides at the intersection of physics-based and statistical forensics with benefits from both sides.

***Index Terms*—Image forensics, splicing detection, color processing, image formation, JPEG compression, downsampling, low quality images**

I. Introduction

IMAGE-based communication became commonplace with the rise of social networks and the broad availability of network-connected acquisition devices. Images effectively transport messages and emotions, and are widely accepted as proof for an event. However, sophisticated image processing software makes it increasingly easy to realistically manipulate images. Manipulations with malicious intent change the message of an image, with the purpose to deceive the viewer.

One important goal of image forensics is to develop methods for detecting image manipulations. Over the past two decades, various methods have been proposed for exposing image forgeries. A general overview can be found in recent books and surveys [1]–[3].

Most of the current research concentrates on statistical methods. Examples are traces from the digital image formation, such as fixed-pattern sensor noise [4] or noise residuals [5], [6]. Other statistical methods search for potential tampering cues such a double JPEG compression [7], [8] or artifacts from computer-generated images [9]–[13]. Another branch is physics-based forensics, which validates for example the lighting direction [14]–[17], lighting color [18]–[20], shadows [21], [22], or perspective constraints [23]–[25].

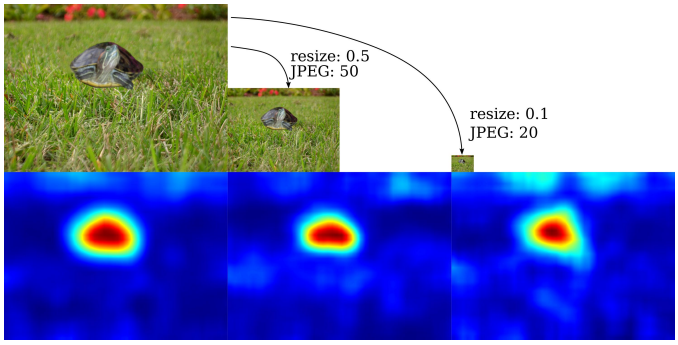


Fig. 1: Spliced tortoise (manipulation from NIMBLE 2017 dataset), and detection result under challenging downsampling and compression factors as they may occur during distribution over the internet. The detection heatmaps in the bottom row are barely affected even by JPEG quality 20 and downsampling to 10% of the original size.

Both families of approaches have unique advantages and limitations. Statistical methods achieve unparalleled performance on high-quality images, while physics-based methods can oftentimes better process strongly compressed or downsampled low-quality images. Statistical methods have the added benefit to allow batch processing on images with arbitrary content, while physics-based methods oftentimes require specific image content [15], [18], and they also oftentimes require manual annotations by an analyst [15], [21], [22].

In this work, we combine concepts from statistical and physics-based methods to harvest benefits from both. We propose a forensic descriptor to characterize the color formation in an image. Color is impacted by scene objects, illumination, and the in-camera processing pipeline. We present a metric space that co-locates similar illumination and in-camera processing artifacts while largely ignoring scene objects.

The distances in this space can be used to detect image splices. One concrete example is shown in Fig. 1, where the tortoise is spliced in the image (top). This insertion can be detected on a heatmap from our proposed method (bottom). The proposed method shares with physics-based methods the benefit of being remarkably robust to compression and downsampling, which is illustrated by the three variants of the same picture in horizontal direction and their associated heatmaps. At the same time, the method can run in a fully automated fashion, and is as such less dependent on user input than classical physics-based methods.

This work was preceded by a conference paper from our group on supervised learning of color differences [20]. That previous work used a complicated acquisition protocol for ground-truth training data with a Macbeth color checker to

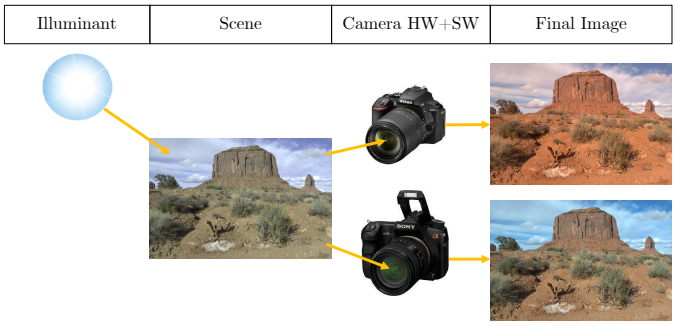


Fig. 2: Illustration of the color image formation. Image colors are determined by the scene illuminant, object reflectances, and the camera color processing in hardware (HW) and software (SW). Our goal is to learn invariance to object reflectances, while preserving sensitivity to differences in illumination and camera processing.

train a regression network. In contrast, this work proposes a *metric space* without any need for Macbeth ground truth data. This enables the use of significantly more data for training with a much larger diversity. The resulting method performs considerably better, and outperforms the state of the art on strongly compressed images.

The main contributions of this work are:

- A practical, scalable method to learn distances in a metric space that represent the similarity of color imaging conditions.
- An extensive performance evaluation that demonstrates the usefulness of this color representation for strongly compressed and downsampled images, where other forensic methods are oftentimes challenged.
- Evaluations on white-point estimation and camera model identification to demonstrate that the learned feature space generalizes to camera identification and illuminant estimation.

The paper is organized as follows: in Section II, we review the related literature. Section III presents the theoretical background, our proposed method, and its application to exposing image manipulations. In Section IV, we evaluate the robustness of the learned embeddings, benchmark our algorithm for forgery detection and localization, and further analyze the transferability of the learned features to other vision tasks. Section V concludes this work.

II. Related Work

Many forensic approaches aim to expose image manipulations by validating a particular consistency property. This property is assumed to be satisfied by pristine images. A violation of this property is considered as an indicator of manipulation. In this spirit, this work proposes a learned color-based descriptor to establish a consistency criterion that jointly includes camera properties, camera settings, and scene properties. We hence review in this section two directions of related works, namely on camera forensics and color-based forensics.

A. Forensic Characterization of the Acquisition Device

The first forensic criteria on camera consistency were formed via analytic models of specific processing steps in the imaging pipeline. For example, manipulations can be exposed by examining color filter array interpolation [26] or lateral chromatic aberration, which particularly applies to low-quality lenses [27]. Arguably the most impactful analytic approach is based on photo-response non-uniformity (PRNU). PRNU is a fixed-noise pattern of the camera sensor that allows for associating an image to its acquisition device [4] and exposing local manipulations [28].

Analytic approaches are appealing to forensic practitioners due to the inherent explainability of observations. However, analytic approaches are difficult to adapt to complicated processing chains. Such complications arise for example on computational images from modern smartphones and with social network post-processing. As a consequence, recent methods learn consistency criteria directly from the data. For example, Bondi *et al.* propose a convolutional neural network (CNN) to predict the acquisition camera from image patches [29]. The CNN is trained on a closed set of known cameras to locate inserted patches from another camera. For manipulation detection on an open set of cameras, Mayer and Stamm train a Siamese network to compare image regions [5]. This work has been further extended to also operate on post-processed images [30]. Cozzolino and Verdoliva propose an image-to-image mapping to obtain a camera model fingerprint of high-frequency artifacts [6]. This fingerprint directly exposes statistical inconsistencies in manipulated regions.

These learning-based methods have in common that they exploit the camera provenance of an image, and hence require training data with source camera labels. Huh *et al.* relax the need for labeled data via a metadata-based self-supervised learning approach [31]. Here, pair-wise patch consistency is learned, and manipulations can be exposed by aggregating all pair-wise consistencies. The metadata enables the network to implicitly learn a discriminator which also uses traces other than camera provenance. Our approach is in some sense similar to that work, but we consider the *scene provenance* of patches instead of image metadata. Additionally, we explicitly include the effect of in-camera color processing as a consistency criterion. Note that in-camera color processing is different from other camera provenance-based approaches since it does not require data with provenance labels, which potentially allows to use very large datasets for training.

B. Forensic Characterization of Scene Colors

The first forensic criteria on scene color consistency were formed via analytic, physics-based models of the illuminant color. Gholap and Bora validate the presence of a global scene illuminant via the intersection of dichromatic lines on specular objects [32]. Riess and Angelopoulou calculate an illuminant map from local illuminant color estimates, which can subsequently be analyzed by an expert [33].

As both methods require manual interaction, they are not applicable to a fully automatic analysis. Carvalho *et al.* limit the analysis of illuminant maps to human faces to automate

the analysis [18]. Stanton *et al.* use a white point in the image as reference to assess the consistency of local illuminant estimates with a CNN [19]. Pomari *et al.* train a CNN for fake image classification on illuminant maps [34]. Here, the manipulation is localized by propagating the classifier decision backwards through the CNN to the input layer. However, such discriminatively trained algorithms require pristine *and* manipulated images for training, which makes them susceptible to unseen manipulations.

In an earlier conference paper, we propose a two-stage approach to expose manipulations from inconsistent color imaging conditions, specifically from the illuminant color and camera color processing [20]. A CNN is trained to locally estimate a color descriptor and a classifier for pairwise consistency. However, the supervised training of the CNN depends on ground-truth images with Macbeth color charts, which are extremely tedious to acquire. Hence, the amount of available training data is severely limited, which implicitly also limits the method performance.

The work in this paper significantly improves our earlier work, by approaching color inconsistencies from a different angle: We propose an unsupervised deep metric learning approach that *disentangles illumination and camera-based color traces from scene content using arbitrary RAW images*. This allows to train a neural network on considerably larger datasets to improve the performance.

As a sidenote, some computer vision methods apply related problem statements to different applications. For example, Lalonde and Efros investigate color distributions of composite images to assess their perceived realism [35]. Gao *et al.* model the in-camera color pipeline for visually plausible insertion of full 3-D computer graphics models [36]. Our method aims at more subtle, imperceptible color mismatches of inserted content that might deceive the viewer. As such, our method can potentially help to raise the bar for methods that primarily aim at visually plausible splicings.

III. Proposed Method

This Section consists of five parts. In Sec. III-A, we introduce a model for color image formation to derive our approach. Section III-B relates this model to the tasks of estimating the illuminant color and camera identification. Section III-C introduces the creation of training and validation data of identical scenes imaged with different color pipelines. Section III-D presents the details of how to practically learn the metric color features. Section III-E presents the bottom-up aggregation of patch-wise decisions for exposing image manipulations.

A. Theoretical Background

a) Image Formation: We denote a position in the image domain $\mathcal{X} \subset \mathbb{R}^2$ as $\boldsymbol{x} \in \mathcal{X}$ and a wavelength in the electromagnetic spectrum $\Lambda \subset \mathbb{R}_+$ as $\lambda \in \Lambda$. Let $\boldsymbol{I}^{c,s}(\boldsymbol{x})$ denote the RGB intensities of an image $\boldsymbol{I}^{c,s} : \mathcal{X} \to \mathbb{R}^3_+$ at position $\boldsymbol{x}$, where $s \in \mathcal{S}$ denotes one specific scene out of the set of all scenes $\mathcal{S}$, and $c \in \mathcal{C}$ denotes one specific camera color processing pipeline out of the set of all pipelines $\mathcal{C}$. Then, the image formation is modeled as (compare [37])

$$\boldsymbol{I}^{c,s}(\boldsymbol{x}) = \boldsymbol{\Omega}_c \left(\int_\Lambda e_s(\lambda, \boldsymbol{x}) \cdot r_s(\lambda, \boldsymbol{x}) \cdot \boldsymbol{c}_c(\lambda) \, \mathrm{d}\lambda \right) \quad . \tag{1}$$

Here, the illuminant spectral density $e_s : \Lambda \times \mathcal{X} \to \mathbb{R}_+$ models the potentially spatially varying power density of the scene illuminants in dependence of λ. The term $r_s = \tilde{r}_s \cdot \tilde{m}_s : \Lambda \times \mathcal{X} \to \mathbb{R}_+$ denotes the reflectance. It consists of the spatially varying spectral reflectance $\tilde{r}_s : \Lambda \times \mathcal{X} \to \mathbb{R}_+$ that determines what fraction of the light striking a surface is reflected for a given λ, and $\tilde{m}_s : \mathcal{X} \to \mathbb{R}_+$ that determines the ratio of reflected light in dependence of the surface geometry. The vector-valued spectral camera sensitivity $\boldsymbol{c}_c : \Lambda \to \mathbb{R}^3_+$ determines the fraction of light of wavelength λ that passes through the red, green, and blue (RGB) color filters. The subsequent and potentially nonlinear *in-camera* color processing operations, such as white balancing or color space transformation, are modeled by the vector-valued function $\boldsymbol{\Omega}_c : \mathbb{R}^3_+ \to \mathbb{R}^3_+$. Note that, although the argument of $\boldsymbol{\Omega}_c$ depends on $\boldsymbol{x}$, the actual function $\boldsymbol{\Omega}_c$ is spatially invariant, and only determined by the camera itself. The positions $\boldsymbol{x} = (\frac{m}{M}, \frac{n}{N})$ are discretized by the sensor with $m \in \{0, \dots, M-1\}$ and $n \in \{0, \dots, N-1\}$. A sampling point $\boldsymbol{x}$ yields a 3-dimensional RGB-vector $\boldsymbol{I}^{c,s}(\boldsymbol{x}) \in \mathbb{R}^3_+$.

With this definition, a discretized image $\boldsymbol{I}^{c,s}$ is represented as element of $\mathbb{R}_+^{M \times N \times 3}$. An image patch $\boldsymbol{p}_i^{c,s} \in \mathbb{R}^{M' \times N' \times 3}$ collects all RGB-samples in a rectangular neighborhood around its center $\boldsymbol{x}_i$ with $M' \leq M$ and $N' \leq N$. We denote by $\mathcal{I}^{c,s}$ the set of all indices of patches from image $\boldsymbol{I}^{c,s}$, and use $i \in \mathcal{I}^{c,s}$ as index for individual patches.

b) Learning Invariances and Covariances: Our fundamental assumption is that a manipulated image contains parts from multiple sources, and hence exhibits traces of different color imaging conditions $(e_s, \boldsymbol{\Omega}_c, \boldsymbol{c}_c)$. These traces are captured in local features that are learned by a CNN, i.e., from image patches $\boldsymbol{p}_i^{c,s}$.

The proposed CNN is represented by the parametric family of functions $f_{\boldsymbol{\theta}} : \mathcal{P} \to \mathcal{M}$ with parameters $\boldsymbol{\theta}$. It maps patches $\boldsymbol{p}_i^{c,s} \in \mathcal{P} \subseteq \mathbb{R}^{M' \times N' \times 3}$ to embeddings $\boldsymbol{y}_i^{c,s} = f_{\boldsymbol{\theta}}(\boldsymbol{p}_i^{c,s}) \in \mathcal{M} \subseteq \mathbb{R}^q$ in a metric space $(\mathcal{M}, d)$. We use the metric $d : \mathcal{M} \times \mathcal{M} \to [0, 1]$, where

$$d(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1}) = \frac{1}{2} \cdot (1 - s(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1})) \quad . \tag{2}$$

Note that we simplify the notation by omitting the indices c and s when they are not required for disambiguation. In Eqn. (2), $s : \mathcal{M} \times \mathcal{M} \to [-1, 1]$ denotes the cosine similarity

$$s(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1}) = \frac{\boldsymbol{y}_{i_0}^\top \boldsymbol{y}_{i_1}}{\|\boldsymbol{y}_{i_0}\| \cdot \|\boldsymbol{y}_{i_1}\|} \quad . \tag{3}$$

The normalization discards differences in length, such that only the angular distance is measured. This allows similar vectors to cluster radially.

Our goal is to learn an embedding space where the embedding distances $d(\boldsymbol{y}_{i_0}^{c_0,s_0}, \boldsymbol{y}_{i_1}^{c_1,s_1})$ characterize the degree of agreement or discrepancy in two color imaging conditions

$(e_{s_0}, \boldsymbol{\Omega}_{c_0}, \boldsymbol{c}_{c_0})$ and $(e_{s_1}, \boldsymbol{\Omega}_{c_1}, \boldsymbol{c}_{c_1})$. To this end, we observe that in Eqn. (1), the camera properties $(\boldsymbol{\Omega}_c, \boldsymbol{c}_c)$ are independent from the scene s, and the scene properties (e_s, r_s) are independent from the camera c. Using these independencies, we argue that the following two *additional* conditions suffice to learn the desired embedding:

1) $f_{\boldsymbol{\theta}}$ maps two patches from the same image closer than two patches from the same scene but different camera color pipelines. Formally, if $c_0, c_1 \in \mathcal{C}$, $c_0 \neq c_1$ denote two different camera color pipelines, and $s \in S$ one arbitrary scene, then

$$\forall i_0, i_1 \in \mathcal{I}^{c_0,s} \;\; \forall i_2 \in \mathcal{I}^{c_1,s} : \\ d(\boldsymbol{y}_{i_0}^{c_0,s}, \boldsymbol{y}_{i_1}^{c_0,s}) < d(\boldsymbol{y}_{i_0}^{c_0,s}, \boldsymbol{y}_{i_2}^{c_1,s}) \;\; . \tag{4}$$

2) $f_{\boldsymbol{\theta}}$ maps two patches from the same image closer than two patches from different scenes regardless of the camera pipeline. Formally, if $c_0, c_1 \in \mathcal{C}$ are any two camera color pipelines, and $s_0, s_1 \in S$, $s_0 \neq s_1$ are two different scenes, then

$$\forall i_0, i_1 \in \mathcal{I}^{c_0,s_0} \;\; \forall i_2 \in \mathcal{I}^{c_1,s_1} : \\ d(\boldsymbol{y}_{i_0}^{c_0,s_0}, \boldsymbol{y}_{i_1}^{c_0,s_0}) < d(\boldsymbol{y}_{i_0}^{c_0,s_0}, \boldsymbol{y}_{i_2}^{c_1,s_1}) \;\; . \tag{5}$$

Both of these conditions enforce small pairwise distances within the same image. For such within-image patch pairs, the camera parameters $(\boldsymbol{\Omega}_c, \boldsymbol{c}_c)$ are identical, but the spatially variant scene parameters $(e_s(\boldsymbol{x}), r_s(\boldsymbol{x}))$ differ if $\boldsymbol{x}_{i_0} \neq \boldsymbol{x}_{i_1}$. These pairs encourage the CNN to learn an *invariance* against within-scene reflectance variations $\Delta r_s(\,\cdot\,, \Delta\boldsymbol{x})$ and within-scene illuminant variations $\Delta e_s(\,\cdot\,, \Delta\boldsymbol{x})$.

The inequality in Eqn. (4) encourages large pairwise distances between patches from the same scene s and different camera pipelines c_0 and c_1. We implicitly use the within-scene invariance to reflectance $\Delta r_s(\,\cdot\,, \Delta\boldsymbol{x})$ due to varying image content. Hence, Eqn. (4) induces a *covariance* with between-camera variations of the parameters $\Delta\boldsymbol{\Omega}_{\Delta c}$, i.e., only the variations in the camera parameters contribute to this larger distances.

The inequality in Eqn. (5) encourages long distances between clusters of embeddings from different scenes. Using again implicitly the within-scene invariance to reflectance $\Delta r_s(\,\cdot\,, \Delta\boldsymbol{x})$, Eqn. (5) induces a *covariance* with respect to between-scene illuminant variations $\Delta e_{\Delta s}$ with the additional (mild) assumption that illuminant variations across scenes $\Delta e_{\Delta s}$ are larger than illuminant variations within the same scene $\Delta e_s(\,\cdot\,, \Delta\boldsymbol{x})$.

In summary, both conditions together aim to learn embeddings with invariance against within-scene illuminant and reflectance variations, and covariance with between-scene illuminant variations and between-camera parameter variations. These two conditions are enforced during training: they dictate the boolean ground truth matrices $\boldsymbol{B}_k$, that determine whether distances between pairs of embeddings are minimized or maximized, respectively, as described in Sec. III-D. The distances between these embeddings characterize discrepancies in the color imaging conditions $(e_s, \boldsymbol{\Omega}_c, \boldsymbol{c}_c)$.

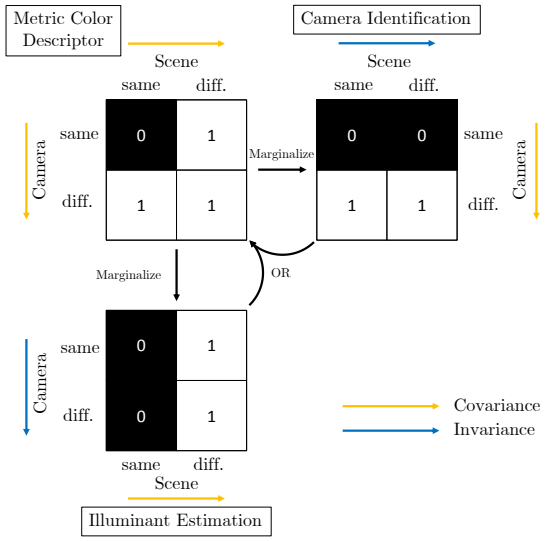


Fig. 3: The proposed approach in relation to camera identification and illuminant estimation. "0", resp. "1", in the (pairwise boolean ground truth) matrices indicates that distances for the respective patch pairings are minimized, resp. maximized. The learned features inherit the covariances with respect to the camera pipeline and the scene. The tasks of camera identification and illuminant estimation can be regarded as marginalization of the proposed approach, while the proposed approach can be regarded as unification (disjunction of boolean ground truth matrices) of the two.

B. Relation to Camera Identification and Illuminant Color Estimation

The proposed metric learning approach is closely related to the tasks of camera identification and illuminant estimation. Camera identification requires features that only depend on the camera ("covariance" w.r.t. the camera), but not on the image content ("invariance" w.r.t. the scene) [38]. Conversely, illuminant estimation requires features that only depend on the scene ("covariance" w.r.t. the scene), but not on the camera ("invariance" w.r.t. the camera) [39].

Figure 3 illustrates that the proposed metric color descriptor unites the covariances for both tasks. This makes it possible to expose forgeries by discrepancies of either of the two quantities. At the same time, it is also possible to "re-use" the metric space for both tasks individually: features for camera identification and illuminant estimation can be obtained via marginalization of the space, by replacing one covariance with an invariance. We experimentally demonstrate this property in Sec. IV-D. There, we apply the learned features to camera identification and also to illuminant estimation.

C. Datasets and Training/Test/Validation Splits

The training requires images of identical scenes imaged with different camera pipelines. To obtain such images, we post-process RAW images to final images with various parameter combinations that affect the color processing. Each RAW image constitutes one scene.

After removing broken and completely over- or underexposed images, we use a total of 5997 RAW images from the RAISE database [40], 4998 RAW images from the MIT-Adobe FiveK dataset [41], 1632 RAW images from the dataset

by Nam and Kim [42], and 645 RAW images crawled from `raw.pixls.us`.

The RAW-to-final-image conversion is performed with `rawpy`, a Python wrapper for `LibRaw`. The camera pipelines differ in the applied white balancing and color space transformations. 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". These parameters yield a total of $|\mathcal{C}| = 12$ camera pipelines per RAW image. The resulting images differ in their color appearance, as can be seen for the input images in Fig. 4. For some combinations of scenes and acquisition devices, a subset of the pipelines can lead to almost identical results, which is the reason for the use of the filter criterion in Eqn. (8).

We use the preset training/validation/test split for the dataset by Nam and Kim [42]. The remaining data is split by scenes with a ratio of 0.8, 0.1, 0.1, such that the content of validation and test images has not been seen during training. This yields a total of $|\mathcal{S}^{\text{train}}| = 10494$, $|\mathcal{S}^{\text{val}}| = 1463$, and $|\mathcal{S}^{\text{test}}| = 1315$ scenes for training, validation and test, and thus $|\mathcal{S}^{\text{train}}| \cdot |\mathcal{C}| = 125928$, $|\mathcal{S}^{\text{val}}| \cdot |\mathcal{C}| = 17556$, and $|\mathcal{S}^{\text{test}}| \cdot |\mathcal{C}| = 15780$ images in total.

The training and validation process operates on image patches. Training and validation patches without variation in content, and patches that are severely under- or overexposed are excluded to ensure sufficient color variation. More specifically, we exclude patches with $\min \boldsymbol{p}_i = \max \boldsymbol{p}_i$, and only allow patches with few overexposed pixels

$$\sum_{k=0}^{2} \mathbb{1}\left(\left(\sum_{m=0}^{M'-1}\sum_{n=0}^{N'-1} \mathbb{1}(\boldsymbol{p}_i[m,n,k] > \rho_{\text{hi}})\right) > \delta_{\text{hi}}\right) \leq N_{\text{hi}} \tag{6}$$

and few underexposed pixels

$$\sum_{k=0}^{2} \mathbb{1}\left(\left(\sum_{m=0}^{M'-1}\sum_{n=0}^{N'-1} \mathbb{1}(\boldsymbol{p}_i[m,n,k] < \rho_{\text{lo}})\right) > \delta_{\text{lo}}\right) \leq N_{\text{lo}} \tag{7}$$

with indicator function $\mathbb{1}(\,\cdot\,)$ and intensity thresholds ρ_{hi}, ρ_{lo}, δ_{hi}, δ_{lo}, N_{hi} and N_{lo}. These equations ensure that at most N_{hi} color channels exhibit more than δ_{hi} overexposed pixels with intensity larger than ρ_{hi}. Underexposed pixels are analogously treated with N_{lo}, δ_{lo} and ρ_{lo}. Furthermore, when patches are paired with the same scene but different camera pipelines, we ensure a minimum distance δ_{Lab} of both patches in *Lab*-color space:

$$\forall c_1 \in \mathcal{C}_k^s,\ \ c_1 \neq c_0 : \sum_{m=0}^{M'-1}\sum_{n=0}^{N'-1} \frac{\sqrt{\sum_{k=0}^{2} (\tilde{\boldsymbol{p}}_i^{c_0,s}[m,n,k] - \tilde{\boldsymbol{p}}_i^{c_1,s}[m,n,k])^2}}{M'N'} \geq \delta_{\text{Lab}}\ , \tag{8}$$

where $\mathcal{C}_k^s$ is the set of all cameras selected for scene s in batch k, see Sec. III-D, and $\tilde{\boldsymbol{p}}_i$ denotes patch $\boldsymbol{p}_i$ in *Lab*-color space. This constraint prevents the attempt to learn differences between color pipelines that do not manifest in differences of the color patch.

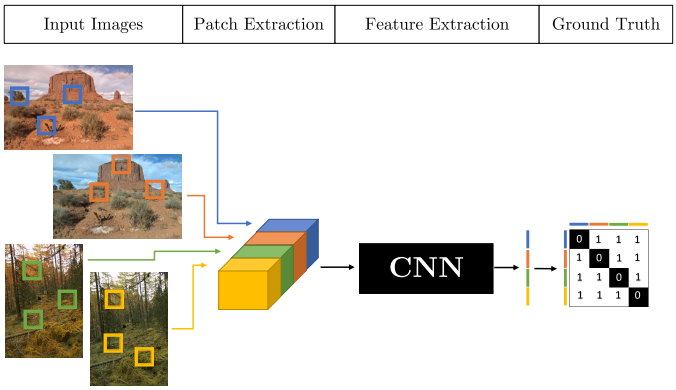


Fig. 4: Illustration of the proposed training approach. Patches from the same scene *and* same camera pipeline are similar ("0" in boolean ground truth matrix), while patches from different camera pipelines *or* different scenes are dissimilar ("1").

D. Learning Metric Color Features

The proposed CNN is based on the ResNet-50 architecture [43], with input patch size $M' \times N' = 128 \times 128$ and weights pretrained on ImageNet [44]. The output layer is replaced with a linear dense layer with $q = 64$ units, where the weights are initialized with Glorot Uniform Initialization [45]. The theoretical model is incorporated into the training of this network, which is described in the remainder of this Section.

The parameters $\boldsymbol{\theta}$ of the CNN $f_{\boldsymbol{\theta}}$ are optimized to satisfy the conditions in Eqns. (4) and (5) to obtain the desired invariances and covariances. To this end, the CNN is presented training data in mini-batches $\boldsymbol{P}_k \in \mathbb{R}^{N_b \times M' \times N' \times 3}$ indexed by k. Each mini-batch is stack of N_b patches $\boldsymbol{p}_i$. The CNN maps a mini-batch $\boldsymbol{P}_k$ to the output $\boldsymbol{Y}_k \in \mathbb{R}^{N_b \times q}$ of N_b embeddings $\boldsymbol{y}_i$. Given $\boldsymbol{Y}_k$, we compute the matrix $\boldsymbol{D}_k \in [0,1]^{N_b \times N_b}$ of pairwise distances with $\boldsymbol{D}_k[i_0, i_1] = d(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1})$ and indexing operator $[\,\cdot\,]$. We further define boolean ground truth matrices $\boldsymbol{B}_k^\star \in \{0,1\}^{N_b \times N_b}$, i.e., masks indicating similar ($\boldsymbol{B}_k^\star[i_0, i_1] = 0$) and dissimilar ($\boldsymbol{B}_k^\star[i_0, i_1] = 1$) pairings for indices i_0 and i_1. The parameters $\boldsymbol{\theta}$ are iteratively updated after each mini-batch, to continuously decrease the distances of similar embeddings while continuously increasing the distances of dissimilar embeddings, as indicated by $\boldsymbol{B}_k^\star$.

During each training epoch, we split $\mathcal{S}^{\text{train}}$ in disjoint subsets $\{\mathcal{S}_k^{\text{train}}\}_k$ with $|\mathcal{S}_k^{\text{train}}| := N_b^{\mathcal{S}} \leq |\mathcal{S}^{\text{train}}|$. One such subset is used per minibatch k. For each $s \in \mathcal{S}_k^{\text{train}}$, we select a random subset $\mathcal{C}_k^s \subset \mathcal{C}$ of different camera pipelines with $|\mathcal{C}_k^s| := N_b^{\mathcal{C}} < |\mathcal{C}|$. Finally, from each of the resulting $N_b^{\mathcal{S}} \cdot N_b^{\mathcal{C}}$ images, we randomly extract $N_b^{\mathcal{P}}$ patches, which yields a total of $N_b^{\mathcal{S}} \cdot N_b^{\mathcal{C}} \cdot N_b^{\mathcal{P}} := N_b$ patches per mini-batch. Note that the patch exclusion criteria from the previous subsection (under-/overexposure, color variation) have been applied before. After all scenes have been processed in an epoch, the order of the scenes is randomly permuted before starting the next epoch to support the training process.

The elements of the boolean ground truth mask $\boldsymbol{B}_k^\star[i_0, i_1]$ are set in agreement with the constraints for $\boldsymbol{y}_{i_0}^{c_0,s_0}$ and $\boldsymbol{y}_{i_1}^{c_1,s_1}$ in Sec. III-A: patch pairs from within the same image image ($s_0 = s_1 \wedge c_0 = c_1$) are considered similar, and obtain a logical 0 in the ground truth mask. Patch pairs are dissimilar if they

stem from different scenes ($s_0 \neq s_1$) or from the same scene but from different camera pipelines ($s_0 = s_1 \wedge c_0 \neq c_1$). These pairs obtain logical 1, indicating dissimilarity. The resulting boolean ground truth matrix $\boldsymbol{B}_k^\star = \boldsymbol{B}^\star = \text{const.}$ is a block matrix. On the diagonal are $N_b^\mathcal{S} \cdot N_b^\mathcal{C}$ block matrices of shape $N_b^\mathcal{P} \times N_b^\mathcal{P}$ that consist only of zeros. All remaining $N_b^\mathcal{S} \cdot N_b^\mathcal{C} \cdot (N_b^\mathcal{S} \cdot N_b^\mathcal{C} - 1)$ off-diagonal blocks of the same shape consist only of ones. Figure 4 provides an overview of the training approach for $N_b^\mathcal{S} = 2$, $N_b^\mathcal{C} = 2$ and $N_b^\mathcal{P} = 3$.

To measure and optimize the performance of the CNN, we employ the histogram loss $\mathcal{L}_{\boldsymbol{\theta}}^{\text{hist}}$ by Ustinova and Lempitsky [46]. This loss function measures the probability that a similar pair of embeddings has a larger distance $d(\boldsymbol{y}_{i_0}^{c_0,s_0}, \boldsymbol{y}_{i_1}^{c_1,s_1})$ than a dissimilar pair. In [46], this so-called *probability of the reverse* is expressed using the probability densities p^+, p^- of the pairwise similarties of similar and dissimilar pairs, which are approximated using histograms, denoted h_r^+, h_r^-, respectively. Following Ustinova and Lempitsky, we estimate h_r^+ and h_r^- at equidistant nodes with $t_0 = -1$ and $t_{R-1} = 1$, and we fix $R = 26$. Each similarity is herein attributed to the two neighboring nodes with weights proportional to the distances between samples and nodes. Note that we only consider distances $d(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1})$ for $i_0 > i_1$ to avoid identical and duplicate pairs. With the cumulative density function $\Phi_r^+ = \sum_{q=0}^{r} h_q^+$, the loss function is obtained as

$$\mathcal{L}_{\boldsymbol{\theta}}^{\text{hist}} = \sum_{r=0}^{R-1} h_r^- \Phi_r^+ \ . \tag{9}$$

The number R of bins of the histograms is the only parameter of $\mathcal{L}_{\boldsymbol{\theta}}^{\text{hist}}$; however, it has negligible effect on the outcome [46].

We further employ the so-called global orthogonal regularizer $\mathcal{R}_{\boldsymbol{\theta}}^{\text{orth}}$, proposed by Zhang *et al.* [48], to enforce dissimilar embeddings $\boldsymbol{y}_{i_0}^{c_0,s_0}$ and $\boldsymbol{y}_{i_1}^{c_1,s_1}$ with $c_0 \neq c_1 \vee s_0 \neq s_1$ to spread out in the target domain $\mathcal{M}$. This is achieved by regularizing the mean M_1 and second moment M_2 of the distribution of pairwise distances of dissimilar pairs towards the moments $M_1^\mathcal{U} = 0$ and $M_2^\mathcal{U} = q^{-1}$ of a uniform distribution on the unit sphere $\mathbb{S}^{q-1}$. With $\mathcal{I}_-^\star = \{(i_0, i_1) \mid i_0 > i_1 \wedge \boldsymbol{B}^\star[i_0, i_1] = 1\}$, the moments are computed as

$$M_1 = \frac{1}{|\mathcal{I}_-^\star|} \sum_{(i_0,i_1)\in\mathcal{I}_-^\star} s(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1}) \ , \tag{10}$$

$$M_2 = \frac{1}{|\mathcal{I}_-^\star|} \sum_{(i_0,i_1)\in\mathcal{I}_-^\star} s(\boldsymbol{y}_{i_0}, \boldsymbol{y}_{i_1})^2 \ , \tag{11}$$

such that the regularizer is obtained as

$$\mathcal{R}_{\boldsymbol{\theta}}^{\text{orth}} = M_1^2 + \max(0, M_2 - \frac{1}{q}) \ . \tag{12}$$

In total, the CNN is trained by iteratively minimizing

$$\mathcal{L}_{\boldsymbol{\theta}} = \mathcal{L}_{\boldsymbol{\theta}}^{\text{hist}} + \eta \cdot \mathcal{R}_{\boldsymbol{\theta}}^{\text{orth}} \tag{13}$$

for some weighting factor η.

During training, we artificially degrade and randomly augment the image patches to improve the generalization and robustness of the network. Prior to filtering for under-/overexposure and color variation in Eqn. (6) to Eqn. (8), each image is resized with constant aspect ratio, such that the larger image dimension is 1536 pixels. Each extracted patch is further augmented via random horizontal or vertical flipping, random rotation and shearing, resizing, and JPEG compression. Rotation and shearing is done with angles in the range of $[-5°, 5°]$. Resizing is done with factors in the range of $[0.95, 1.05]$. A patch is JPEG compressed with a probability of 0.5, with a random quality level in the range $[50, 100]$. These resampling and recompression steps remove much of the high-frequent statistical information, which forces the CNN to learn more robust traces.

The CNN's generalization is measured for identical validation mini-batches throughout all epochs. For this, we use the validation set $\mathcal{S}^{\text{val}}$ filtered for under-/overexposure and color variation as described in Subsec. III-C, but without any patch degradation or augmentation.

The number of similar and dissimilar patch pairs per batch are uneven. To nevertheless measure the separability of the two distributions, the validation performance in the experiments is measured with the area under the ROC curve (ROC AUC) of the distributions of similar and dissimilar patches. Based on validation loss, we select the optimal parameters $\boldsymbol{\theta}^\star$ for manipulation analysis, which is presented in greater detail in Sec. IV-A.

E. Application to Forgery Detection and Localization

We create a heatmap from the trained CNN $f_{\boldsymbol{\theta}^\star}$ for manipulation detection and localization. Here, *detection* denotes a binary yes/no statement whether an image is spliced. *Localization* denotes a segmentation of the spliced region within an image. The specific processing steps are illustrated in Fig. 5 and described below.

A test image $\boldsymbol{I}^{\text{test}} \in \mathbb{R}_+^{M\times N\times 3}$ is first resized with constant aspect ratio such that its larger dimension is 1536 pixels. The following steps are to

1) Extract patches $\boldsymbol{p}_i$ with stride 32, $i \in \{0, \ldots, N^{\text{test}} - 1\}$. This yields N^{test} patches. For typical image aspect ratios, $N^{\text{test}} \approx 10^3$.
2) Compute the CNN embedding $\boldsymbol{y}_i = f_{\boldsymbol{\theta}^\star}(\boldsymbol{p}_i)$ for each patch $\boldsymbol{p}_i$.
3) Compute the medoid $\boldsymbol{\mu}$ of the embeddings,
$$\boldsymbol{\mu} = \underset{\boldsymbol{y}\in\{\boldsymbol{y}_i\}_i}{\text{argmin}} \sum_{i=0}^{N^{\text{test}}-1} d(\boldsymbol{y}, \boldsymbol{y}_i) \ , \tag{14}$$
to obtain a reference point for consistent imaging conditions. Relate each embedding $\boldsymbol{y}_i$ to that reference $\boldsymbol{\mu}$ by computing an inconsistency score $\gamma_i = d(\boldsymbol{\mu}, \boldsymbol{y}_i) \in [0, 1]$.
4) Project the scores γ_i back to the image locations of the corresponding patch $\boldsymbol{p}_i$ to create a heatmap $\tilde{\mathcal{H}}$. In this map, larger values indicate a higher manipulation likelihood.

Finally, the heatmap $\tilde{\mathcal{H}}$ is resized to the original test image dimensions to obtain the heatmap $\mathcal{H} \in [0, 1]^{M\times N}$. Note that during this analysis, we apply the criteria in Eqn. (6) to Eqn. (8), and assign $\gamma_i = 0$ to all patches that were filtered out prior to the analysis.

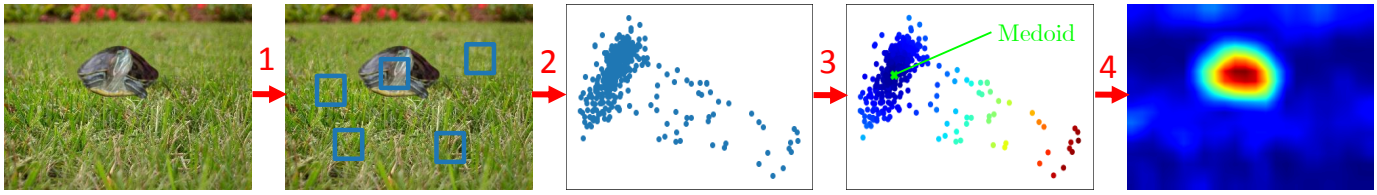


Fig. 5: Application of the CNN embedding to manipulation localization. The distance of each image patch w.r.t. to the medoid of all patches is measured in the learned embedding space (distances color-coded; embeddings projected to $\mathbb{R}^2$ using Multidimensional Scaling [47] for visualization), and then aggregated back into the image space to create a heatmap of manipulation likelihoods.

For forgery *detection*, we compute the average over the heatmap as an image global manipulation score

$$\Gamma = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \mathcal{H}[m,n] \ , \tag{15}$$

with $0 \leq \Gamma \leq 1$.

We also investigate an alternative variant of transforming the pairwise distance matrix $\boldsymbol{D}^{\text{test}}$ to patchwise scores γ_i by using the computationally more complex MeanShift aggregation [49], which is also used in [31].

IV. Experimental Results

In this section, we first report the concrete training parameters and practical training aspects. The subsequent evaluation consists of multiple parts. First, we demonstrate the robustness of the learned embeddings to degraded image qualities. Then, we compare our algorithm with state-of-the-art approaches for manipulation localization and detection on various datasets and image qualities. Finally, we demonstrate the dual nature of the embeddings by applying the metric space to camera identification and white-point estimation.

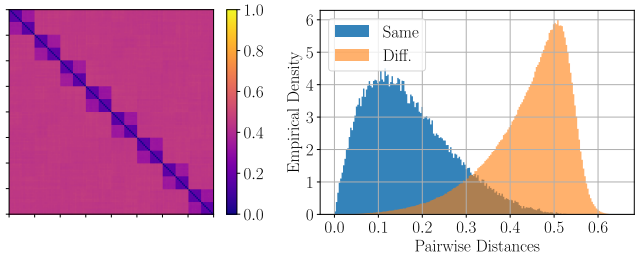
A. Training and Evaluation of the CNN

The hyperparameters are summarized in Tab. I. The patch size is set to $M' \times N' = 128 \times 128$, trading off resolution and number of patches for training. The parameters for the filter criteria in Eqn. (6) and Eqn. (7) are set to $\rho_{\text{hi}} = 252$, $\rho_{\text{lo}} = 5$ for intensities $0 \leq \boldsymbol{p}_i[m,n,k] \leq 255$, and further $\delta_{\text{hi}} = \delta_{\text{lo}} = 0.3M'N'$, $N_{\text{hi}} = 2$ and $N_{\text{lo}} = 3$. For Eqn. (8), we set $\delta_{\text{Lab}} = 5$ with intensities in *Lab*-color space as $0 \leq \tilde{\boldsymbol{p}}_i[m,n,k] \leq 255$. For these parameters, Eqns. (6) and (7) on average exclude 3.3% of patches, that are either over- or underexposed. Equation (8) further excludes 32.8% of patches whose counterparts from different color pipelines are very similar. The above parameter values are used throughout all training, validation and testing runs, except for the experiments depicted in Fig. 7b, Fig. 7c and Fig. 7d, where we investigate the effect of different values for δ_{Lab}.

The tuning of all remaining hyperparameters, as well as the choice of the loss function, have been performed on the validation dataset. The mini-batches for training and validation are created with $N_b^{\mathcal{S}} = 8$, $N_b^{\mathcal{C}} = 2$ and $N_b^{\mathcal{P}} = 8$, i.e., a batchsize of $N_b = 128$. The weight in the loss function in Eqn. (13) is set to $\eta = 0.5$. To optimize the CNN parameters $\boldsymbol{\theta}$, we use the Adam optimizer [50] with learning rate $\alpha = 10^{-4}$ and moments $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

TABLE I: Summary of hyperparameters

Category	Parameter	Value	Comment / Description
Patch size	M', N'	128	patch size: $M' \times N'$
Filter criteria (Eqns. (6)–(8))	ρ_{lo}	5	$0 \leq \boldsymbol{p}_i[m,n,k] \leq 255$
	ρ_{hi}	252	
	δ_{lo}, δ_{hi}	$0.3M'N'$	relative to patch size
	δ_{Lab}	5	$0 \leq \tilde{\boldsymbol{p}}_i[m,n,k] \leq 255$
Batch configuration	$N_b^{\mathcal{S}}$	8	#scenes per batch
	$N_b^{\mathcal{C}}$	2	#cameras per batch
	$N_b^{\mathcal{P}}$	8	#patches per image
	N_b	128	$N_b = N_b^{\mathcal{S}} N_b^{\mathcal{C}} N_b^{\mathcal{P}}$
Loss function	η	0.5	Regularizer weight
Optimizer	α	10^{-4}	Learning rate
	β_1	0.9	Moments
	β_2	0.999	



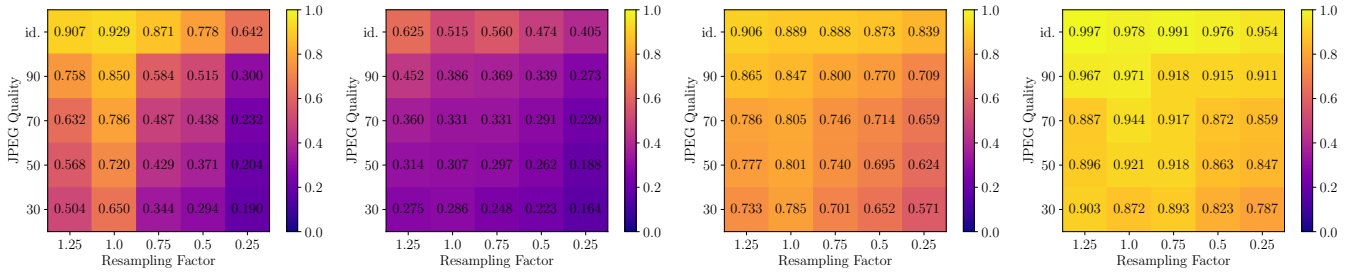
(a) Pairwise distance matrix (b) Distributions of pairwise distances

Fig. 6: Validation performance of the CNN. a) Matrix of pairwise distances, averaged over all validation mini-batches ($N_b = 128$). b) Empirical distributions of pairwise distances for same (blue) and different (orange) imaging conditions.

Whenever the validation performance does not improve for at least 20 epochs, α is decreased by a factor of 10. We apply early stopping once the validation loss is not decreasing anymore, to obtain the final parameters $\boldsymbol{\theta}^\star$. This protocol lead to 149 epochs for CNN training.

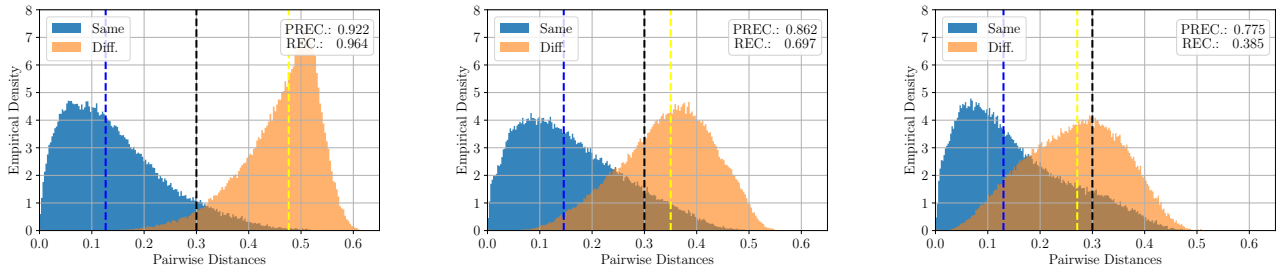
On a machine with 32 CPUs with 128 GB RAM and an NVIDIA GeForce GTX 1080 GPU with 12 GB RAM, one epoch runs for about 40 minutes in our implementation with Python and Keras [51]. Hence, a full training run requires about 100 hours. The final validation loss is $\mathcal{L}^{\text{val}} = 0.0657$.

Figure 6 visualizes the validation performance. Figure 6a shows the pairwise distance matrix averaged over all validation mini-batches. The average distances for pairs of embeddings from identical imaging conditions (blocks of size $N_b^{\mathcal{P}} \times N_b^{\mathcal{P}} = 8 \times 8$ on the diagonal) are consistently lower than the distances for different imaging conditions (off-diagonal blocks). Pairs of different scenes exhibit larger distances than pairs that only differ in the camera pipeline. Hence, different



(a) Different scenes (b) Same scenes, $\delta_{\text{Lab}}^{\min} = 0$ (c) Same scenes, $\delta_{\text{Lab}}^{\min} = 20$ (14.7% of patches) (d) Same scenes, $\delta_{\text{Lab}}^{\min} = 50$ (0.883% of patches)

Fig. 7: Separation of the empirical distributions of the pairwise embedding distances for patches from images of similar and dissimilar imaging conditions for various post-processing operations, measured in terms of True Positive Rate at a fixed False Alarm Rate of 5%. a) 50 pairs of images of different scenes. b)-d) 50 image pairs where one scene each has been imaged with two different camera pipelines for increasing minimum *Lab*-distances.



(a) $f = 1.0$, $Q = \text{id}$. (b) $f = 0.75$, $Q = 70$ (c) $f = 0.25$, $Q = 30$

Fig. 8: Empirical distributions of the pairwise embedding distances for similar and dissimilar imaging conditions from 50 pairs of different scenes with increasing post-processing strength from (a) to (c). Precision and recall are measured for a decision threshold fixed to 0.3 (black dashed line). The colored dashed lines indicate the median of the corresponding distribution. Decreasing quality moves the distribution of distances for dissimilar pairs to smaller values, while the distribution for similar pairs is barely affected.

scenes are easier separable than different camera pipelines for identical scenes. Figure 6b shows the empirical distributions of pairwise distances for same and different imaging conditions, aggregated over all validation mini-batches. The distribution for similar pairs concentrates on small distances around 0.1, whereas the distribution for dissimilar pairs peaks at about 0.5. This corresponds to orthogonal embeddings, as enforced by the regularizer in Eqn. (12). The validation ROC AUC is 0.967, indicating a good separation of the distributions.

B. Robustness of the Learned Embeddings

This experiment shows the robustness against resizing and JPEG recompression, which are common post-processing operations on images from social networks. We randomly select 100 test scenes from $\mathcal{S}^{\text{test}}$, and per scene a random camera pipeline to obtain 100 test images.

Post-processing is applied for each combination of resizing factors $f \in \{1.25, 1.0, 0.75, 0.5, 0.25\}$ and JPEG quality levels $Q \in \{\text{id.}, 90, 70, 50, 30\}$, where $f = 1.0$ implies no resizing, id. ("idempotent") implies no JPEG recompression. Then, each image is resized such that the larger dimension is 1536 pixels, and embeddings are calculated for 50 random pairs of images and 50 random patches per image. We then compute distributions of within-image pairwise distances and between-image pairwise distances to represent similar and dissimilar imaging conditions.

We measure the separation of the distributions in terms of True Positive Rate (TPR) at a fixed False Alarm Rate (FAR) of 5%, which we denote $\text{TPR}_{5\%}$. The results for embeddings on different scenes are shown in Fig. 7a. For images with no post-processing ($f = 1.0$, $Q = \text{id.}$), the distributions separate well with a $\text{TPR}_{5\%}$ of 0.929. For resampling factors down to 0.5 and JPEG quality down to 50, the $\text{TPR}_{5\%}$ is still 0.371 or higher. Extreme downsampling and compression with $f = 0.25$ and $Q = 30$ still yields an $\text{TPR}_{5\%}$ of 0.190, which is remarkably high given these strong degradations.

The more challenging case of identical scene but different camera pipelines is reported in Fig. 7b, Fig. 7c, and Fig. 7d. The three evaluation protocols only differ in the minimum color difference of the evaluated image pairs, compare Eqn. (8), the remaining evaluation protocol is unchanged. From left to right, patches are compared with a minimum *Lab* color difference that is arbitrarily small ($\delta_{\text{Lab}}^{\min} = 0$), at least $\delta_{\text{Lab}}^{\min} = 20$ (14.2% of test patches), and at least $\delta_{\text{Lab}}^{\min} = 50$ (0.883% of test patches).

For arbitrary similar color distributions $\delta_{\text{Lab}}^{\min} = 0$, $\text{TPR}_{5\%}$ ranges between 0.625 and 0.262 for resampling factors down to $f = 0.5$ and JPEG compression down to $Q = 50$. Extreme degradations with $f = 0.25$ and $Q = 30$ still achieve a $\text{TPR}_{5\%}$ of 0.164. It is encouraging that this worst-case configuration still maintains good separability significantly above guessing chance, which is equal to the fixed FPR of 5%. The easier

case of a minimum color difference $\delta_{\mathrm{Lab}}^{\min} = 20$ between the pipelines exhibits a $\mathrm{TPR}_{5\%}$ above 0.57 throughout all post-processing parameters. Even larger color differences $\delta_{\mathrm{Lab}}^{\min} = 50$ can be almost perfectly separated, with $\mathrm{TPR}_{5\%}$ above 0.78 across all degradations. These results show that larger differences in camera pipelines lead to a better separability. Overall, the learned embeddings are remarkably robust to even strong resampling and JPEG compression.

We now consider the behavior of the pairwise distance distributions for pairs from different images (compare Fig. 7a) for increasing post-processing strengths with respect to a fixed threshold ϑ. A patch pair is classified as dissimilar ("positive"), if their distance in the learned metric space is larger than ϑ, and as similar ("negative") otherwise. The quality of the split can be measured with precision (PREC) and recall (REC),

$$\mathrm{PREC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \ , \tag{16}$$

$$\mathrm{REC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \ , \tag{17}$$

calculated from the true positives TP, true negatives TN, false positives FP and false negatives FN. Based on the empirical distance distributions on the validation data, see Fig. 6b, we fix $\vartheta = 0.3$.

In case of no post-processing (Fig. 8a), the distributions are split well by $\vartheta = 0.3$ with PREC 0.922 and REC 0.964. For $f = 0.75$ and $Q = 70$ (Fig. 8b), PREC is still at 0.862, while REC is 0.697. For extreme downsampling and compression with $f = 0.25$ and $Q = 30$, PREC still amounts to 0.775, while REC is reduced to 0.385.

This behavior can be attributed to a shift of the distribution of positives to lower values with increasing post-processing, while the distribution negatives remains quite stable, as can be seen from the median of the respective distributions in Fig. 8. As a consequence, more positives are classified negative, while the number of false positives remains low. This experiment shows that while the distributions can still be separated well for strong post-processing (compare Fig. 7), for optimal splitting the threshold needs to be adapted depending on the strength of the operations. However, note that splicing localization only requires relative pairwise distances. Hence, determining a globally optimal threshold is only necessary for splicing detection, not localization.

C. Comparison for Splicing Detection and Localization

This Section consists of five parts. First, we present the benchmark datasets and the algorithms for comparison. Then, we evaluate the localization performance by presenting quantitative and qualitative results, which is the primary focus of this method. We finally report detection performances in comparison to related work.

a) Benchmark Datasets: We use a total of four datasets listed in Tab. II. Three datasets have been previously published, namely "DSO-1" [18], "Aligned Scenes" [20] and 'In-The-Wild" [31]. "DSO-1", and "Aligned Scenes" support evaluation of manipulation localization and detection. "In-the-Wild" contains no pristine images, and can hence only assess localization performance. "DSO-1" contains splices of people. "Aligned Scenes" consists of splices where rectangular image blocks are replaced by an image block with the same scene content, but recorded with a different camera. "In-The-Wild" contains a collection of online images with splicing and other manipulations, and manually annotated ground truth.

TABLE II: Overview of the benchmark datasets. The second column details the number of pristine, resp. manipulated images *per* quality level, i.e., the total number of test images per dataset is obtained as product with the number of quality levels.

Name	#Prist. / #Manip.	#Qualities	Tasks
In-The-Wild [31]	0 / 201	3	Loc.
DSO-1 [18]	100 / 100	3	Loc. / Det.
Aligned Scenes [20]	166 / 166	11	Loc. / Det.
SplicedColorPipeline	200 / 200	3	Loc. / Det.

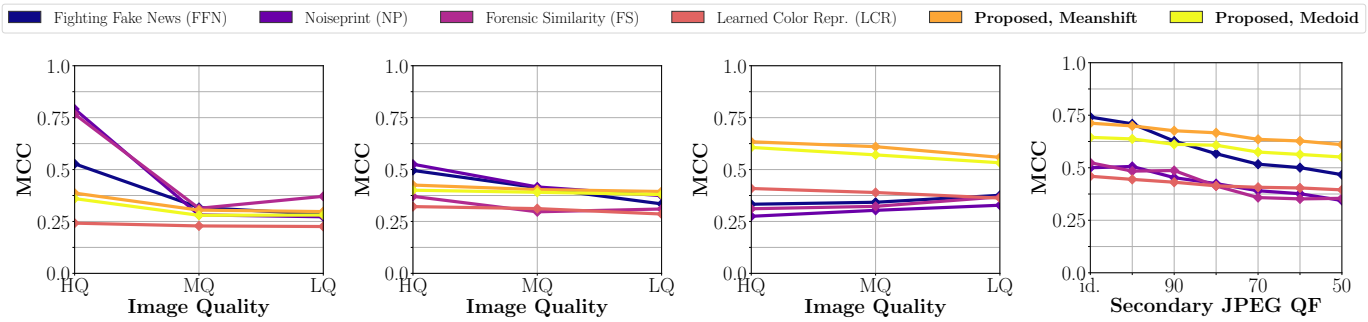
We further create a fourth dataset called "SplicedColor-Pipeline" to evaluate detection and localization for the specific case of different camera pipelines. The dataset consists of 200 pristine images, consisting of randomly selected scenes from $\mathcal{S}^{\mathrm{test}}$, each developed with a randomly selected camera pipeline. 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 [52] with scale parameter 10 and $\sigma = 0.5$. We replace a region such that the average *Lab*-distance between the original and inserted region 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 is first downsampled and then recompressed with JPEG qualities from 100 down to 10 in steps of 10 [20]. 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.

b) Algorithms for Comparison: The proposed method is evaluated for medoid-based scores (MED), and MeanShift-based aggregation (MSA), see Sec. III-E. We benchmark our algorithm with other works not requiring manipulated training images. In particular, we compare against "Noiseprint", (NP) [6], "Fighting Fake News" (FFN) [31], "Forensic Similarity" (FS) [30] and "Learned Color Representations" (LCR) [20].

NP learns an image-to-image mapping, and enforces output distances based on the agreement of the cameras that recorded the images. FFN uses a Siamese CNN and derives labels from image metadata. FS is based on a Siamese network that classifies the camera and post-processing consistency of image patches. LCR learns a color descriptor in a supervised fashion, and checks the consistency of this cue throughout an image.

For the former two algorithms, we take the available im-



(a) DSO-1 [18] (b) In-The-Wild [31] (c) SplicedColorPipeline (d) Aligned Scenes [20]

Fig. 9: Performance for splicing *localization* using the Matthews Correlation Coefficient on various datasets. The proposed method (orange/yellow) compares particularly well on very low-quality images (see text for details).

plementations from their official repositories. For FS, we use the available trained CNN in the variant with input patch sizes 128×128. After filtering out patches with inadequate entropy [30] from a test image, we aggregate pairwise predicted similarities of that CNN to a heatmap using MeanShift as in [31]. For LCR, we use our own implementation, and aggregate pairwise similarity scores in the same way using MeanShift to obtain heatmaps.

c) Splicing Localization: Localization performance is measured with averaged scores over all images per dataset. We use Matthews Correlation Coefficient (MCC), defined as

$$\mathrm{MCC} = \frac{\mathrm{TP}\cdot\mathrm{TN} - \mathrm{FP}\cdot\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}\,, \tag{18}$$

For each image, we use the decision threshold that maximizes the MCC score [6], [31]. Further, as the algorithms cannot distinguish whether foreground or background are inserted, we evaluate heatmap $\mathcal{H}$ and its inverse $1-\mathcal{H}$, and use the better result [6], [31]. Our method is tailored to discrepancies in color imaging conditions and low quality images. However, for better comparison we also show results on general splices and high quality images. The results are shown in Fig. 9.

On the DSO-1 dataset (Fig. 9a), the best results are obtained for the purely statistics-based approaches NP (HQ: 0.791) and FS (HQ: 0.767). The proposed method with MSA achieves a performance of 0.387 for HQ images. The proposed method is challenged by the fact that both the background and the spliced part of the image are in many cases recorded with the same camera and with flash light as dominant light source [18], which violates our assumption that the color formation differs. Surprisingly, all methods quickly deteriorate with decreasing image quality, which may be due to degradation-induced removal of tell-tale manipulation artifacts.

On the In-The-Wild dataset (Fig. 9b), NP performs best for HQ images (0.524), followed by FFN (0.496) and the proposed approach with MSA (0.425). The performance of NP and FFN decreases with increasing degradations (NP LQ: 0.364, FFN LQ: 0.325). The proposed method remains robust (MSA MQ: 0.403, LQ MSA: 0.394), and performs best on low-quality images.

On the SplicedColorPipeline dataset (Fig. 9c), the proposed method (MSA) performs best throughout all qualities with MCC 0.634 and 0.559 for MQ and LQ, respectively. With a large margin, the second best performance is obtained by LCR (HQ: 0.409, MQ: 0.363).

On the Aligned Scenes dataset (Fig. 9d), the proposed method achieves the best results for JPEG qualities of 90 (0.676) and below.

The proposed method outperforms some state-of-the-art approaches for HQ images. However, it particularly excells in its robustness against image degradations, where other methods rapidly lose performance. The proposed method performs best on low-quality images, which are notoriously difficult to analyze for statistical approaches. The SplicedColorPipeline and Aligned Scenes datasets specifically benchmark differences in the color formation. The proposed methods is particularly well-suited to detect these traces, for which it outperforms all remaining methods also for high-quality images.

d) Qualitative Localization Results: Figure 10 shows qualitative results for the proposed method with MeanShift aggregation on images from various datasets. In each row, results are shown for increasing image degradations. Although the manipulations are diverse in size, color, and degree of texture, the proposed method successfully localizes the manipulated regions throughout all qualities.

Figure 11 shows qualitative examples of representative failure cases. Here, the color discrepancies between the inserted regions and parts of the background are smaller than within-background differences.

Table III lists the localization performance for the examples in Figs. 10 and 11 in terms of MCC. In the top six rows, corresponding to Fig. 10, the MCC is high, while for failure cases in the bottom three rows (corresponding to Fig. 11), the MCC is rather low.

e) Splicing Detection: The proposed method, and the methods for comparison, are primarily designed for manipulation localization. However, for completeness, we also report results for splicing detection, although we acknowledge that localization-based methods can be expected to perform worse on this task than specialized methods for splicing detection.

For each algorithm, we compute the average over the heatmap according to Eqn. (15) as a per-image manipulation score. As it is common for splicing detection [17], [19], [53], for each quality level the performance is measured via the ROC AUC. The "In-the-Wild" dataset is not part of this

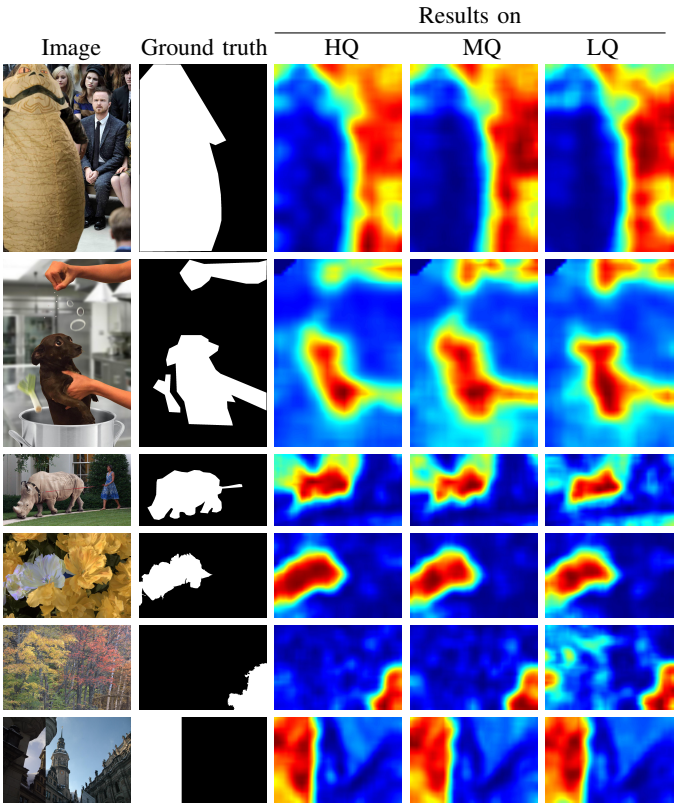


Fig. 10: Qualitative results for manipulation localization using MeanShift. Results barely deteriorate from high-quality (HQ), medium quality (MQ) to low-quality (LQ) images.

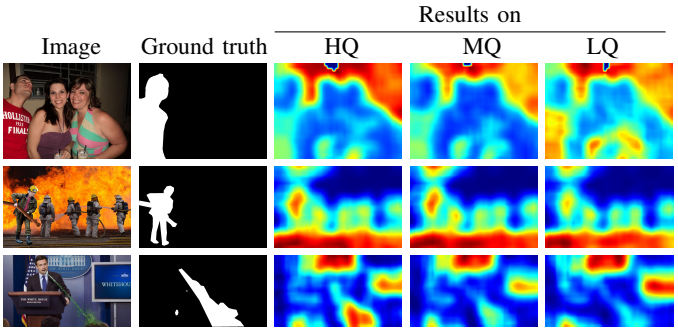


Fig. 11: Failure cases with significant background contrast, evaluated for manipulation localization with MeanShift aggregation.

experiment, as it does not contain any pristine images. The results are shown in Fig. 12. We found that NP exhibits large variations in the values across heatmaps, such that we could not find a competitive detection threshold for this localization method.

Figure 12a shows the results for the DSO-1 dataset. The best performance for HQ images is obtained for the statistics-based FS (HQ: 0.938), where the proposed method achieves an AUC of 0.63.

Figure 12b shows the results for the SplicedColorPipeline dataset. Here, the proposed method performs best throughout all quality levels (HQ MED: 0.656, LQ MED: 0.612). The second best results are obtained for the color-based LCR, followed by FFN that uses various traces. The statistics-based

TABLE III: Performance for splicing localization for the qualitative examples in Figs. 10 and 11 (from top to bottom) in terms of Matthews Correlation Coefficient.

Dataset	Image Basename	HQ	MQ	LQ
In-The-Wild [31]	im31_edit6	0.959	0.958	0.964
	im30_edit2	0.825	0.831	0.853
	2251	0.570	0.583	0.620
SplicedColorPipeline	210uxdvkfw3j0si_1	0.943	0.927	0.946
	2gcbbedjgxy9v7j_1	0.970	0.958	0.837
Align. Sce. [20]	n8tn8fkkx29a	0.998	0.994	0.998
DSO-1 [18]	splicing-02	0.276	0.248	0.228
In-The-Wild [31]	im32_edit7	0.263	0.256	0.232
	5482	0.367	0.217	0.210

NP and FS only achieve random guessing performance, which is expected since the different camera pipelines only modulate color properties, but do not affect other statistical traces. The overall rather low AUC of the proposed method on this dataset can be attributed to difficulty of finding a global threshold for the detection score. The fact that the size of the spliced image regions is not limited during creation of the dataset leads to a large variance in the region sizes, and hence detection scores.

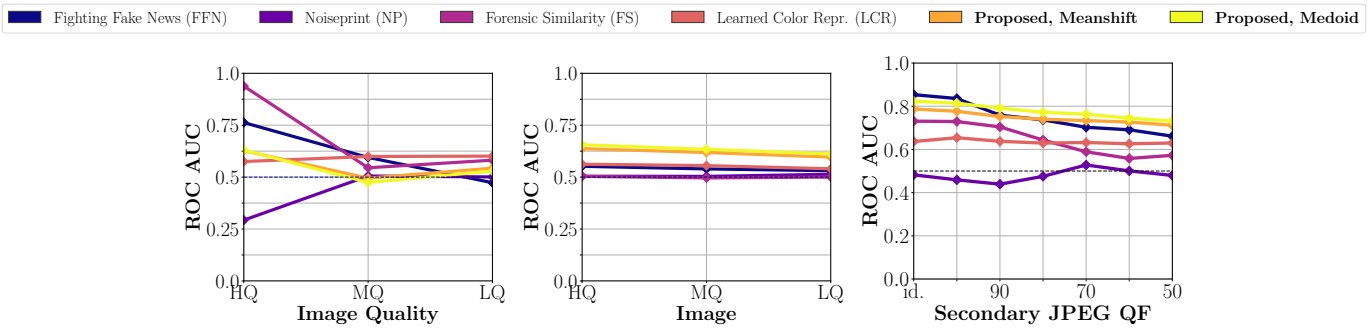
Figure 12c shows the results for the Aligned Scenes dataset. For single compressed and double compressed images with secondary quality 100, the best results are obtained for FFN ("id.": 0.854). For all stronger compressions, the proposed method yields the best results medoid-based scores MED, e.g., for secondary compression quality 90 an AUC of 0.791.

In summary, for manipulations that involve the camera pipeline, the proposed algorithm performs best for non-ideal images with some degradations. For high quality images and other manipulations, the proposed algorithm is outperformed by algorithms that include statistical traces (FFN, FS). For increasing compression, the performance of the proposed approach remains remarkably robust on all datasets except on DSO-1. Also, medoid-based scores generally perform better than MeanShift for detection. We also calculated the true positive rate at a fixed 5% false alarm rate, $\mathrm{TPR}_{5\%}$. The relative performances are almost identical to the AUCs in Fig. 12. At LQ and JPEG 50 settings, example $\mathrm{TPR}_{5\%}$ for our method with medoid-based scores are 3.0%, 10.5%, and 19.3% on the three datasets, which again shows that method works better for splicing localization than detection.

D. Application of the Learned Embeddings to Illuminant Estimation and Camera Identification

This experiment shows that the proposed metric space contains information on camera properties and scene colors. More specifically, Sec. III-B and Fig. 3 indicate that the space can be marginalized for illuminant estimation and for camera identification.

a) Illuminant Estimation: We apply the learned embeddings as features for illuminant estimation. The performance is taken as a surrogate measure for the illuminant information contained in the embeddings. The experiment is performed on the Gehler-Shi dataset [54], [55] using threefold cross-validation. We train a 3-layer neural network, with all hyperparameters tuned on the first validation fold. The images are upscaled to $1,536$ pixels in the larger dimension, and



(a) DSO-1 [18] (b) SplicedColorPipeline (c) Aligned Scenes [20]

Fig. 12: Comparison of splicing *detection* ROC AUC on various datasets. The proposed method is specialized on *localization*. Yet, also for detection it can contribute to the analysis of very low-quality images (see text for details).

TABLE IV: Statistics of the angular error (in degrees) for illuminant estimation. While the proposed method is outperformed by several dedicated illuminant estimators, it exhibits sensitivity to the scene illuminant (see text for details).

Algorithm	**Mean**	**Median**	**Tri-Mean**	**Best** 25%	**Worst** 25%	**Geom. Mean**
SVR [58]	8.08	6.73	7.19	3.35	14.89	7.21
EBG [59]	6.52	5.04	5.43	1.90	13.58	5.40
FOGE [60]	5.33	4.52	4.73	1.86	10.03	4.63
SOGE [60]	5.13	4.44	4.62	2.11	9.26	4.60
NIS [61]	4.19	3.13	3.45	1.00	9.22	3.34
CCC [56]	1.95	1.22	1.38	0.35	4.76	1.40
CNN [62]	1.90	1.12	1.33	0.31	4.84	1.34
FFCC [63]	**1.61**	**0.86**	**1.02**	**0.23**	**4.27**	**1.07**
Ours	5.39	3.21	3.96	1.02	13.31	3.92

gamma-corrected with $\gamma = 0.5$ before patch extraction and computation of the embeddings. The network consists of three dense layers with 64, 32 and 2 units, respectively, to regress the illuminant color in L_uL_v-space, as used in [56]. The weights are initialized with Glorot Uniform initialization [45]. Each layer uses Scaled Exponential Linear Units (SeLU) [57] with Keras default values, except for the output layer with no nonlinearity. We use Mean Squared Error loss, and train with the Adam [50] optimizer with moments $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and batches of size 64. The learning rate is set to $\alpha = 10^{-4}$ and halved if there is no improvement for two consecutive epochs, and we apply Early Stopping.

The L_uL_v-space predictions $\tilde{w}_i \in [0,1]^2$ for patch i are converted to RGB-space, $w_i \in [0,1]^3$, see [56, Eqn. 6]. Then, the RGB-space prediction w of an image is computed as median over all patch predictions, and compared to the ground truth $w^\star$ using angular error

$$\varepsilon = \frac{180^\circ}{\pi} \cdot \cos^{-1}\left(\frac{w^\top w^\star}{\|w\|\|w^\star\|}\right) \quad . \tag{19}$$

As a reference, we compare against selected dedicated illuminant estimation methods: Support Vector Regression (SVR) [58], Edge-based Gamut (EBG) [59], First (FOGE) and Second (SOGE) Order Grey Edge [60], Natural Image Statistics (NIS) [61], a deep learning-based method (CNN) [62], Convolutional Color Constancy (CCC) [56] and Fast Fourier Color Constancy [63]. Table IV lists statistics on the angular errors over all test images and folds, where the last column is the geometric mean of the other five statistics, compare [56], [63]. While the best results are achieved by the recent method FFCC, the learned embeddings outperform dedicated model-based approaches (SVR, EBG, FOGE, SOGE). This suggests that our CNN indeed extracts color information to separate different imaging pipelines.

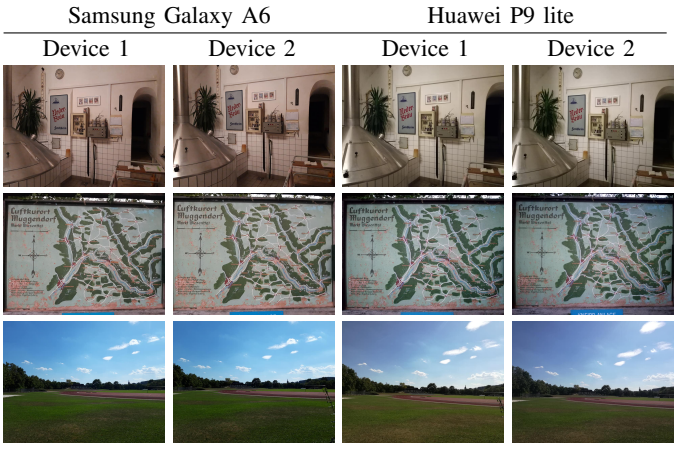
b) Camera Identification: We illustrate the sensitivity of the learned embeddings to camera model traces. For this task, we use images from the recent Forchheim Image Database [64]. This dataset contains images of various scenes, each imaged by different smartphone devices. We compare the learned embeddings for images of identical scenes. These scenes are recorded with two Samsung Galaxy A6 devices and two Huawei P9 lite devices, and are shown in Fig. 13a. Embeddings are calculated for non-overlapping patches from images in *original* quality (confer [64]), i.e., without any post-processing. Then, we apply t-SNE [65] to visualize the high-dimensional embeddings in 2-D, which is shown in Fig. 13b.

Throughout these scenes, the embeddings from devices of the same model tend to cluster, while embeddings from different models are to some extent separated. This distribution indicates that the learned metric space exhibits sensitivity to camera *model* traces, while largely suppressing *device*-specific variations. This is in agreement with our color imaging model leveraged for training, since color formation is in general common to all devices from a model, which underlines the relation of our approach to camera identification.

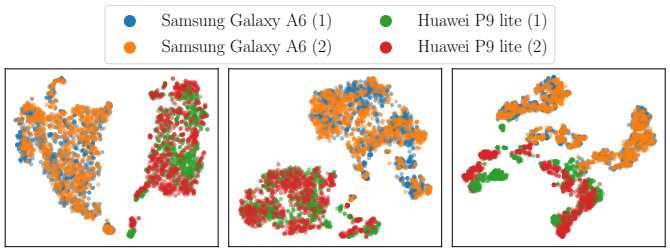
V. Conclusion

We propose a method to learn versatile metric color embeddings that characterize the color consistency of images. Manipulated images can then be exposed by locating image regions with large distances in the learned metric space. We experimentally demonstrate the robustness against the two frequently encountered post-processing measures resizing and JPEG recompression. We further show that the learned embeddings contain information transferable to illuminant estimation and information characterizing discrepancies of camera pipelines.

The proposed method is particularly well-suited for low quality images that have been subject to heavy post-processing. In this case, the color-based analysis is barely affected by



(a) Images of the same scene recorded by different devices in FODB [64]. Scene IDs in FODB from top to bottom: 60, 45, 1.



(b) Scatter plots of the embeddings for patches extracted from the above images, with one color per device, and one plot (from left to right) per scene, i.e., per row (from top to bottom) in Fig. 13a.

Fig. 13: Illustration of the sensitivity of the learned metric space to camera model traces. a) Each row shows images of the same scene photographed by two devices per model. Images from FODB [64]. b) Scatter plots of the learned embeddings for patches extracted from the images in (a), with dimensionality reduction by t-SNE [65].

the loss of high-frequency image content that is central to many other forensic algorithms. As such, we hope that the method can complement a forensic tool set for cases where other algorithms are difficult to apply.

The limitations of our approach include cases where the color variations within pristine parts of images are large, manipulation types that do not affect color distributions, or forgeries involving image-global color adaptations that do not introduce spatial inconsistencies.

ACKNOWLEDGMENT

This material is based on research sponsored by the Air Force Research Laboratory and the Defense Advanced Research Projects Agency under agreement number FA8750-16-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and the Defense Advanced Research Projects Agency or the U.S. Government.

REFERENCES

[1] A. Piva, "An Overview on Image Forensics," *ISRN Signal Processing*, 2013.

[2] H. Farid, *Photo Forensics*. MIT Press, 2016.

[3] L. Verdoliva, "Media Forensics and Deepfakes: An Overview," *arXiv preprint*, 2020.

[4] J. Lukás, J. Fridrich, and M. Goljan, "Detecting Digital Image Forgeries Using Sensor Pattern Noise," in *Security, Steganography, and Watermarking of Multimedia Contents VIII*, 2006.

[5] O. Mayer and M. C. Stamm, "Learned Forensic Source Similarity for Unknown Camera Models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 2012–2016.

[6] D. Cozzolino and L. Verdoliva, "Noiseprint: A CNN-Based Camera Model Fingerprint," *IEEE Trans. Information Forensics and Security*, vol. 15, pp. 144–159, 2020.

[7] I. Amerini, T. Uricchio, L. Ballan, and R. Caldelli, "Localization of JPEG Double Compression Through Multi-Domain Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 1865–1871.

[8] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro, "Aligned and Non-Aligned Double JPEG Detection Using Convolutional Neural Networks," *Journal of Visual Communication and Image Representation*, pp. 153–163, 2017.

[9] S. Lyu, "Deepfake Detection: Current Challenges and Next Steps," in *IEEE International Conference on Multimedia & Expo Workshops*, 2020, pp. 1–6.

[10] X. Xuan, B. Peng, W. Wang, and J. Dong, "On the Generalization of GAN Image Forensics," in *Chinese Conference on Biometric Recognition*, 2019, pp. 134–141.

[11] H. Li, B. Li, S. Tan, and J. Huang, "Identification of Deep Network Generated Images Using Disparities in Color Components," *Signal Process.*, 2020.

[12] M. Barni, K. Kallas, E. Nowroozi, and B. Tondi, "CNN Detection of GAN-Generated Face Images Based on Cross-Band Co-occurrences Analysis," *arXiv preprint*, 2020.

[13] N. Bonettini, P. Bestagini, S. Milani, and S. Tubaro, "On the Use of Benford's Law to Detect GAN-Generated Images," *arXiv preprint*, 2020.

[14] M. K. Johnson and H. Farid, "Exposing Digital Forgeries in Complex Lighting Environments," *IEEE Trans. Information Forensics and Security*, pp. 450–461, 2007.

[15] E. Kee and H. Farid, "Exposing Digital Forgeries From 3-D Lighting Environments," in *IEEE International Workshop on Information Forensics and Security*, 2010, pp. 1–6.

[16] B. Peng, W. Wang, J. Dong, and T. Tan, "Optimized 3D Lighting Environment Estimation for Image Forgery Detection," *IEEE Transactions on Information Forensics and Security*, pp. 479–494, 2016.

[17] F. Matern, C. Riess, and M. Stamminger, "Gradient-Based Illumination Description for Image Forgery Detection," *IEEE Trans. Information Forensics and Security*, pp. 1303–1317, 2020.

[18] T. J. de Carvalho, C. Riess, E. Angelopoulou, H. Pedrini, and A. de Rezende Rocha, "Exposing Digital Image Forgeries by Illumination Color Classification," *IEEE Trans. Information Forensics and Security*, pp. 1182–1194, 2013.

[19] J. Stanton, K. Hirakawa, and S. McCloskey, "Detecting Image Forgery Based On Color Phenomenology," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 138–145.

[20] B. Hadwiger, D. Baracchi, A. Piva, and C. Riess, "Towards Learned Color Representations for Image Splicing Detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 8281–8285.

[21] Q. Liu, X. Cao, C. Deng, and X. Guo, "Identifying Image Composites Through Shadow Matte Consistency," *IEEE Trans. Information Forensics and Security*, pp. 1111–1122, 2011.

[22] E. Kee, J. F. O'Brien, and H. Farid, "Exposing Photo Manipulation With Inconsistent Shadows," *ACM Trans. Graph.*, pp. 28:1–28:12, 2013.

[23] M. Iuliani, G. Fabbri, and A. Piva, "Image Splicing Detection Based on General Perspective Constraints," in *IEEE International Workshop on Information Forensics and Security)*, 2015, pp. 1–6.

[24] B. Peng, W. Wang, J. Dong, and T. Tan, "Image Forensics Based on Planar Contact Constraints of 3D Objects," *IEEE Transactions on Information Forensics and Security*, pp. 377–392, 2017.

[25] X. Yang, Y. Li, and S. Lyu, "Exposing Deep Fakes Using Inconsistent Head Poses," in *IEEE International Conference on Acoustics, Speech and Signal Processing, *, 2019, pp. 8261–8265.

[26] A. C. Popescu and H. Farid, "Exposing Digital Forgeries in Color Filter Array Interpolated Images," *IEEE Transactions on Signal Processing*, pp. 3948–3959, 2005.

[27] M. K. Johnson and H. Farid, "Exposing Digital Forgeries Through Chromatic Aberration," in *Proceedings of the 8th Workshop on Multimedia & Security*, 2006, pp. 48–55.

[28] M. Chen, J. Fridrich, M. Goljan, and J. Lukás, "Determining image origin and integrity using sensor noise," *IEEE Transactions on information forensics and security*, vol. 3, no. 1, pp. 74–90, 2008.
[29] L. Bondi, L. Baroffio, D. Guera, P. Bestagini, E. J. Delp, and S. Tubaro, "First Steps Toward Camera Model Identification With Convolutional Neural Networks," *IEEE Signal Process. Lett.*, pp. 259–263, 2017.
[30] O. Mayer and M. C. Stamm, "Forensic Similarity for Digital Images," *IEEE Trans. Information Forensics and Security*, pp. 1331–1346, 2020.
[31] M. Huh, A. Liu, A. Owens, and A. A. Efros, "Fighting Fake News: Image Splice Detection via Learned Self-Consistency," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 106–124.
[32] S. Gholap and P. Bora, "Illuminant Colour Based Image Forensics," in *TENCON IEEE Region 10 Conference*, 2008, pp. 1–5.
[33] C. Riess and E. Angelopoulou, "Scene Illumination as an Indicator of Image Manipulation," in *International Workshop on Information Hiding*, 2010, pp. 66–80.
[34] T. Pomari, G. Ruppert, E. R. S. D. Rezende, A. Rocha, and T. Carvalho, "Image Splicing Detection Through Illumination Inconsistencies and Deep Learning," in *IEEE International Conference on Image Processing*, 2018, pp. 3788–3792.
[35] J. Lalonde and A. A. Efros, "Using Color Compatibility for Assessing Image Realism," in *IEEE International Conference on Computer Vision*, 2007, pp. 1–8.
[36] J. Gao, X. Li, L. Wang, S. Fidler, and S. Lin, "Mimicking the In-Camera Color Pipeline for Camera-Aware Object Compositing," *arXiv preprint*, 2019.
[37] A. Gijsenij, T. Gevers, and J. Van De Weijer, "Computational Color Constancy: Survey and Experiments," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2475–2489, 2011.
[38] M. Kirchner and T. Gloe, "Forensic camera model identification," *Handbook of Digital Forensics of Multimedia Data and Devices*, pp. 329–374, 2015.
[39] D. Cheng, D. K. Prasad, and M. S. Brown, "Illuminant Estimation for Color Constancy: Why Spatial-Domain Methods Work and the Role of the Color Distribution," *JOSA A*, pp. 1049–1058, 2014.
[40] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "RAISE: A Raw Images Dataset for Digital Image Forensics," in *ACM Multimedia Systems*, 2015, pp. 219–224.
[41] V. Bychkovsky, S. Paris, E. Chan, and F. Durand, "Learning Photographic Global Tonal Adjustment With a Database of Input / Output Image Pairs," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 97–104.
[42] S. Nam and S. J. Kim, "Modelling the Scene Dependent Imaging in Cameras with a Deep Neural Network," in *IEEE International Conference on Computer Vision*, 2017, pp. 1726–1734.
[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
[44] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A Large-Scale Hierarchical Image Database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
[45] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.
[46] E. Ustinova and V. S. Lempitsky, "Learning Deep Embeddings with Histogram Loss," in *Advances in Neural Information Processing*, 2016, pp. 4170–4178.
[47] J. B. Kruskal, "Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis," *Psychometrika*, pp. 1–27, 1964.
[48] X. Zhang, F. X. Yu, S. Kumar, and S. Chang, "Learning Spread-Out Local Feature Descriptors," in *IEEE International Conference on Computer Vision*, 2017, pp. 4605–4613.
[49] Y. Cheng, "Mean Shift, Mode Seeking, and Clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 790–799, 1995.
[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint*, 2014.
[51] F. Chollet *et al.*, "Keras," https://keras.io, 2015.
[52] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation," *International Journal of Computer Vision*, pp. 167–181, 2004.
[53] D. Cozzolino, G. Poggi, and L. Verdoliva, "Splicebuster: A New Blind Image Splicing Detector," in *2015 IEEE International Workshop on Information Forensics and Security*, 2015, pp. 1–6.
[54] P. V. Gehler, C. Rother, A. Blake, T. P. Minka, and T. Sharp, "Bayesian Color Constancy Revisited," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
[55] L. Shi and B. Funt, "Re-processed Version of the Gehler Color Constancy Dataset of 568 Images," http://www.cs.sfu.ca/ colour/data/.
[56] J. T. Barron, "Convolutional Color Constancy," in *IEEE International Conference on Computer Vision*, 2015, pp. 379–387.
[57] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-Normalizing Neural Networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 971–980.
[58] B. Funt and W. Xiong, "Estimating Illumination Chromaticity via Support Vector Regression," in *Color and Imaging Conference*, 2004, pp. 47–52.
[59] A. Gijsenij, T. Gevers, and J. Van De Weijer, "Generalized Gamut Mapping Using Image Derivative Structures for Color Constancy," *International Journal of Computer Vision*, pp. 127–139, 2010.
[60] J. Van De Weijer, T. Gevers, and A. Gijsenij, "Edge-Based Color Constancy," *IEEE Transactions on Image Processing*, pp. 2207–2214, 2007.
[61] A. Gijsenij and T. Gevers, "Color Constancy Using Natural Image Statistics and Scene Semantics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 687–698, 2010.
[62] W. Shi, C. C. Loy, and X. Tang, "Deep Specialized Network for Illuminant Estimation," in *European Conference on Computer Vision*, 2016, pp. 371–387.
[63] J. T. Barron and Y.-T. Tsai, "Fast Fourier Color Constancy," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 886–894.
[64] B. Hadwiger and C. Riess, "The Forchheim Image Database for Camera Identification in the Wild," in *Pattern Recognition. ICPR International Workshops and Challenges - Proceedings, Part VI*, 2020, pp. 500–515.
[65] L. Van der Maaten and G. Hinton, "Visualizing Data Using t-SNE." *Journal of Machine Learning Research*, pp. 2579–2605, 2008.

Benjamin Hadwiger received the B.Eng. degree in electrical engineering from the Nürnberg Institute of Technology Georg Simon Ohm, Nürnberg, Germany, in 2014. In 2016, he received the M.Sc. degree in computational engineering from the Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany. Currently, he is pursuing the Ph.D. degree in computer science at FAU. His research interests include image forensics, computer vision and machine learning.

Christian Riess received the Ph.D. degree in computer science from the Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany, in 2012. From 2013 to 2015, he was a Postdoc at the Radiological Sciences Laboratory, Stanford University, Stanford, CA, USA. Since 2015, he is the head of the Phase-Contrast X-ray Group at the Pattern Recognition Laboratory at FAU. Since 2016, he is senior researcher and head of the Multimedia Security Group at the IT Infrastructures Lab at FAU, and he received his habilitation from FAU in 2020. He served on the IEEE Information Forensics and Security Technical Committee 2017-2019, and received the IEEE Signal Processing Award in 2017. His research interests include all aspects of image processing and imaging, particularly with applications in image and video forensics, X-ray phase contrast, and computer vision.