Fighting Fake News: Image Splice Detection via Learned Self-Consistency

Minyoung Huh*[1,2] Andrew Liu*[1] Andrew Owens[1] Alexei A. Efros[1]

UC Berkeley[1] Carnegie Mellon University[2]

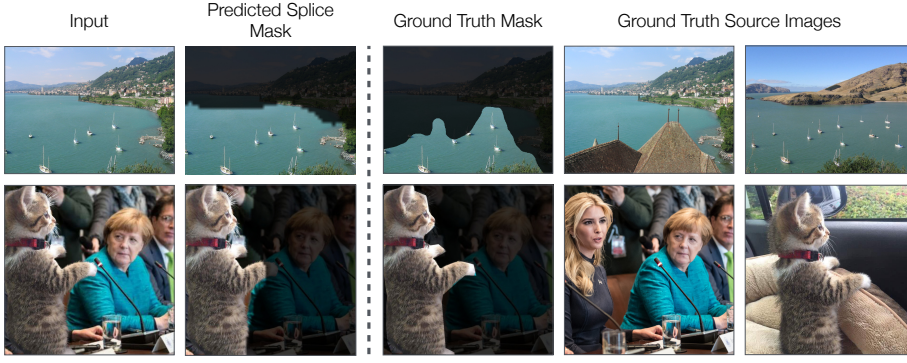

Fig. 1: Our algorithm learns to detect and localize image manipulations (splices), despite being trained only on unmanipulated images. The two input images above might look plausible, but our model correctly determined that they have been manipulated because they lack self-consistency: the visual information within the predicted splice region was found to be inconsistent with the rest of the image. IMAGE CREDITS: automatically created splice from Hays and Efros [1] (top), manual splice from *Reddit* user */u/Name-Albert_Einstein* (bottom).

Abstract. Advances in photo editing and manipulation tools have made it significantly easier to create fake imagery. Learning to detect such manipulations, however, remains a challenging problem due to the lack of sufficient amounts of manipulated training data. In this paper, we propose a learning algorithm for detecting visual image manipulations that is trained only using a large dataset of real photographs. The algorithm uses the automatically recorded photo EXIF metadata as supervisory signal for training a model to determine whether an image is *self-consistent* — that is, whether its content could have been produced by a single imaging pipeline. We apply this self-consistency model to the task of detecting and localizing image splices. The proposed method obtains state-of-the-art performance on several image forensics benchmarks, despite never seeing any manipulated images at training. That said, it is merely a step in the long quest for a truly general purpose visual forensics tool.

Keywords: Visual forensics, image splicing, self-supervised learning, EXIF

*Indicates equal contribution.
Code and additional results can be found on our website.

Fig. 2: **Anatomy of a splice:** a fake image is created by splicing together content from two source images. The insight explored in this paper is that patches from spliced images are typically produced by different imaging pipelines, as indicated by the EXIF meta-data of the two source images. The problem is that in practice, we never have access to these source images.[1]

1 Introduction

Malicious image manipulation, long the domain of dictators and spy agencies, has now become accessible to legions of common Internet trolls and Facebook con-men [2]. With only rudimentary editing skills, it is now possible to create realistic image composites [3, 4], fill in large image regions [1, 5, 6], generate plausible video from speech [7, 8], etc. One might have hoped that these new methods for creating synthetic visual content would be met with commensurately powerful techniques for detecting fakes, but this has not been the case so far.

One problem is that standard supervised learning approaches, which have been very successful for many types of detection problems, are not well-suited for image forensics. This is because the space of manipulated images is so vast and diverse, that it is rather unlikely we will ever have enough manipulated training data for a supervised method to fully succeed. Indeed, detecting visual manipulation can be thought of as an anomaly detection problem — we want to flag anything that is "out of the ordinary," even though we might not have a good model of what that might be. In other words, we would like a method that does not require any manipulated training data at all, but can work in an unsupervised/self-supervised regime.

In this work, we turn to a vast and previously underutilized source of data, image EXIF metadata. EXIF tags are camera specifications that are digitally engraved into an image file at the moment of capture and are ubiquitously available. Consider the photo shown in Figure 2. While at first glance it might seem authentic, we see on closer inspection that a car has been inserted into the scene. The content for this spliced region came from a different photo, shown on the right. Such a manipulation is called an *image splice*, and it is one of the most common ways of creating visual fakes. If we had access to the two source photographs, we would see from their EXIF metadata that there are a number of differences in the imaging pipelines: one photo was taken with an *Nikon* camera, the other with a *Kodak* camera; they were shot using different focal lengths, and

[1] Photo credits: NIMBLE dataset [9] and *Flickr* user James Stave.

saved with different JPEG quality settings, etc. Our insight is that one might be able to detect spliced images because they are composed of regions that were captured with different imaging pipelines. Of course, in forensics applications, we do not have access to the original source images nor, in general, do we even have access to the fraudulent photo's metadata.

Instead, in this paper, we propose to use the EXIF metadata as *supervisory signal* for training a classification model to determine whether an image is *self-consistent* – that is, whether different parts of the same image could have been produced by a single imaging pipeline. The model is self-supervised in that only real photographs and their EXIF meta-data are used for training. A consistency classifier is learned for each EXIF tag separately using pairs of photographs, and the resulting classifiers are combined together to estimate self-consistency of pairs of patches in a novel input image. We validate our approach using several benchmark datasets and show that the model performs better than the state-of-the-art — despite never having seen annotated splices or using handcrafted detection cues.

The main contributions of this paper are: 1) posing image forensics as a problem of detecting violations in learned self-consistency (a kind of anomaly detection), 2) proposing photographic metadata as a free and plentiful supervisory signal for learning self-consistency, 3) applying our self-consistency model to detecting and localizing splices. We also introduce a new dataset of image splices obtained from the internet, and experimentally evaluate which photographic metadata is predictable from images.

2 Related work

Over the years, researchers have proposed a variety of visual forensics methods for identifying various manipulations [2]. The earliest and most thoroughly studied approach is to use domain knowledge to isolate physical cues within an image. Drawing upon techniques from signal processing, previous methods focused on cues such as misaligned JPEG blocks [10], compression quantization artifacts [11], resampling artifacts [12], color filtering array discrepancies [13], and camera-hardware "fingerprints" [14]. We take particular inspiration from the recent work of Agarwal and Farid [15] which exploits a seemingly insignificant difference between imaging pipelines to detect spliced image regions — namely, the way that different cameras truncate numbers during JPEG quantization. While these domain-specific approaches have proven to be extremely useful due to their easy interpretability, we believe that the use of machine learning will open the door to discovering many more useful cues while also producing more adaptable algorithms.

Indeed, recent work has moved away from using *a priori* knowledge and toward applying end-to-end learning methods for solving specific forensics tasks using labeled training data. For example, Salloum et al. [16] propose learning to detect splices by training a fully convolutional network on labeled training data. These learning methods have also been applied to the problem of detecting specific tampering cues, such as double-JPEG compression [17, 18] and contrast enhancement [19]. The most closely related of these methods to ours is perhaps Bondi et al. [20, 21]. This work recognizes camera models from image patches, and proposes to use inconsistencies in camera pre-

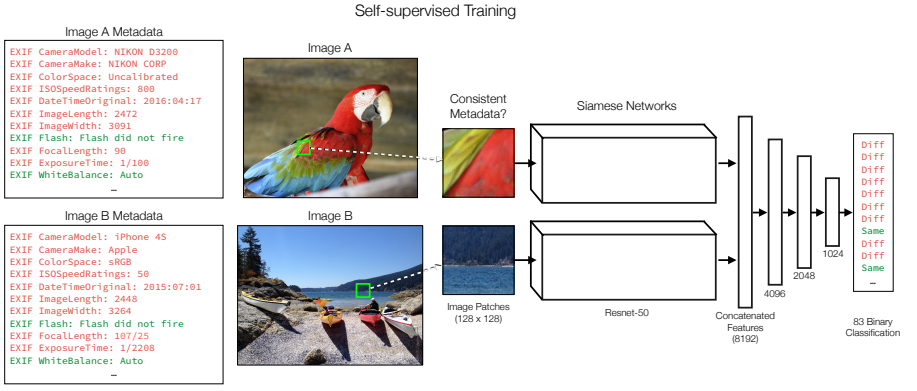

Fig. 3: **Self-supervised training:** Our model takes two random patches from different images and predicts whether they have consistent meta-data. Each attribute is used as a consistency metric during training and testing.

dictions to detect tampering. In concurrent work, Mayer et al. [22] proposed using a Siamese network to predict whether pairs of image patches have the same camera model — a special case of our metadata consistency model (they also propose using this model for splice detection; while promising, these results are very preliminary). There has also been work that estimates whether a photo's semantic content (e.g., weather) matches its metadata [23].

In our work, we seek to further reduce the amount of information we provide to the algorithm by having it learn to detect manipulations without ground-truth annotations. For this, we take inspiration from recent works in self-supervision [24, 25, 26, 27, 28, 29] which train models by solving tasks solely defined using unlabeled data. Of these, the most closely related approach is that of Doersch et al. [25], in which they trained a model to predict the relative position of pairs of patches within an image. Surprisingly, the authors found that their method learned to utilize very subtle artifacts like chromatic lens aberration as a shortcut for learning the task. While imaging noise was a nuisance in their work, it is a useful signal for us — our self-supervised algorithm is designed to learn about properties of the imaging pipeline while ignoring semantics. Our technical approach is also similar to [30], which trains a segmentation model using self-supervision to predict whether pairs of patches co-occur in space or time.

Individual image metadata tags, such as focal length, GPS, hashtags, etc. have long been employed in computer vision as free supervisory signal. A particularly creative use of EXIF metadata was demonstrated by Kuthirummal et al. [31], who used the `CameraModel` tag of a very large image collection to compute per-camera priors such as their non-linear response functions.

Our work is also related to the anomaly detection problem. Unlike traditional visual anomaly detection work, which is largely concerned with detecting unusual semantic events like the presence of rare objects and actions [32, 33], our work needs to find anomalies in photos whose content is designed to be plausible enough to fool humans.

Therefore the anomalous cues we search for should be imperceptible to humans and invariant to the semantics of the scene.

3 Learning Photographic Self-consistency

Our model works by predicting whether a pair of image patches are consistent with each other. Given two patches, $\mathcal{P}_i$ and $\mathcal{P}_j$, we estimate the probabilities $x_1, x_2, ..., x_n$ that they share the same value for each of n metadata attributes. We then estimate the patches' overall consistency, c_{ij}, by combining our n observations of metadata consistency. At evaluation time, our model takes a potentially manipulated test image and measures the consistency between many different pairs of patches. A low consistency score indicates that the patches were likely produced by two distinct imaging systems, suggesting that they originate from different images. Although the consistency score for any single pair of patches will be noisy, aggregating many observations provides a reasonably stable estimate of overall image self-consistency.

3.1 Predicting EXIF Attribute Consistency

We use a Siamese network to predict the probability that a pair of 128×128 image patches shares the same value for each EXIF metadata attribute. We train this network with image patches randomly sampled from $400,000$ *Flickr* photos, making predictions on all EXIF attributes that appear in more than $50,000$ photos ($n = 80$, the full list of attributes can be found in Section A1). For a given EXIF attribute, we discard EXIF values that occur less than 100 times. The Siamese network uses shared ResNet-50 [34] sub-networks which each produce 4096-dim. feature vectors. These vectors are concatenated and passed through four-layer MLP with 4096, 2048, 1024 units, followed by the final output layer. The network predicts the probability that the images share the same value for each of the n metadata attributes.

We observe that training with a typical random sampling procedure will not be successful because: 1) there are some rare EXIF values that will be very difficult to learn, and 2) randomly selected pairs of images are unlikely to have consistent EXIF values just by chance. Therefore, we introduce two types of re-balancing during training: unary and pairwise. For unary re-balancing, we oversample rare EXIF attribute values (e.g. rare camera models). When constructing a mini-batch, we first choose an EXIF attribute and uniformly sample an EXIF value from all possible values of this attribute. For pairwise re-balancing, we make sure that pairs of training images within a mini-batch are selected such that for a given EXIF attribute, half the batch share that value and half do not.

Analysis. Although we train on all common EXIF attributes, we expect the model to excel at distinguishing ones that directly correlate to properties of the imaging pipeline such as `LensMake` [25, 20]. In contrast, arbitrary attributes such as the exact date an image was taken (`DateTimeOriginal`) leave no informative cues in an image. In order to identify predictive metadata, we evaluated our EXIF-consistency model on a dataset of 50K held-out photos and report the individual EXIF attribute accuracy

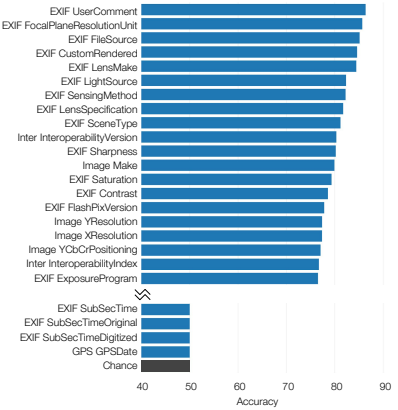

Fig. 4: **EXIF Accuracy:** How predictable are EXIF attributes? For each attribute, we compute pairwise-consistency accuracy on *Flickr* images using our self-consistency model.

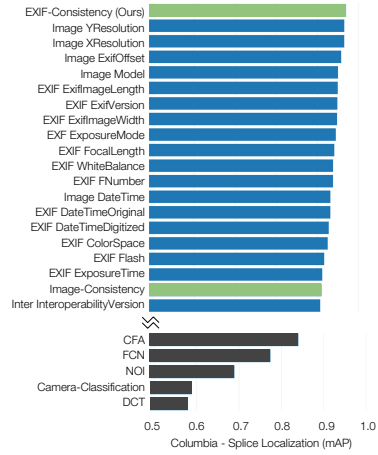

Fig. 5: **EXIF Splice Localization:** How useful are EXIF attributes for localizing splices? We compute individual localization scores on the *Columbia* dataset.

in Figure 4. Because the accuracy is computed on pairwise-balanced batches, chance performance is 50%.

Our model obtains high accuracy when predicting the consistency of attributes closely associated with the image formation process such as `LensMake`, which contains values such as *Apple* and *FUJIFILM*. But more surprisingly, we found that the most predictable attribute is `UserComment`. Upon further inspection, we found that `UserComment` is a generic field that can be populated with arbitrary data. We found that the most frequent values were either binary strings embedded by camera manufacturers or logs left by image processing software. For example, one of the common `UserComment` values, *Processed with VSCOcam*, is added by a popular photo-filtering application. In Section A1, we show the full list of EXIF attributes and their corresponding definitions.

3.2 Post-processing Consistency

Many image manipulations are performed with the intent of making the resulting image look plausible to the human eye: spliced regions are resized, edge artifacts are smoothed, and the resulting image is re-JPEGed. If our network could predict whether two patches are post-processed differently, then this would be compelling evidence for photographic inconsistency. To model post-processing consistency, we add three augmentation operations during training: re-JPEGing, Gaussian blur, and image resizing. Half of the time, we apply the same operations to both patches; the other half of the time, we apply different operations. The parameters of each operation are randomly chosen from an evenly discretized set of numbers. We introduce three additional classification tasks (one per augmentation type) that are used to train the model to predict whether a

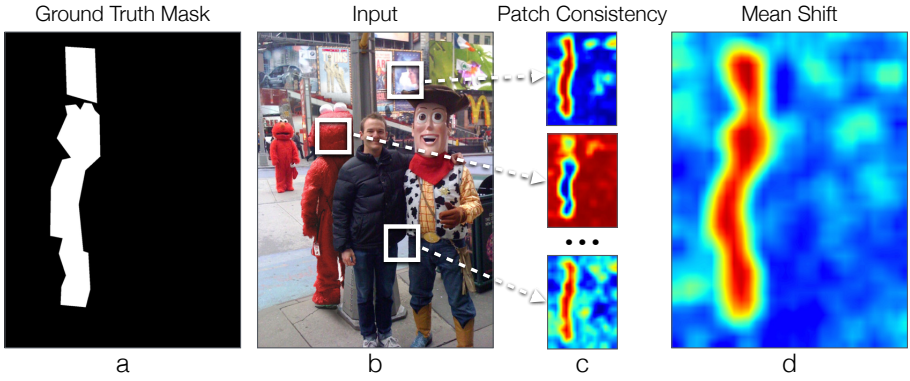

Fig. 6: **Test Time:** Our model samples patches in a grid from an input image (b) and estimates consistency for every pair of patches. (c) For a given patch, we get a consistency map by comparing it to all other patches in the image. (d) We use Mean Shift to aggregate the consistency maps into a final prediction.

pair of patches received the same parameterized augmentation. This increases the number of binary attributes we predict from 80 to 83. Since the order of the post-processing operations matters, we apply them in a random order each time.

We note that this form of inconsistency is orthogonal to EXIF consistency. For example, in the (unlikely) event that a spliced region had exactly the same metadata as the image it was inserted into, the splice could still be detected by observing differences in post-processing.

3.3 Predicting Patch Consistency

Once we have predicted the consistency of a pair of patches for each of our EXIF (plus post-processing) attributes, we would like to estimate the pairs' *overall* consistency c_{ij}. If we were solving a supervised task, then a natural choice would be to use spliced regions as supervision to predict, from the n EXIF-consistency predictions, the probability that the two patches belong to different regions. Unfortunately, we do not have spliced images to train on. Instead, we use a self-supervised proxy task: we train a simple classifier to predict, from the EXIF consistency predictions, whether the patches come from the same image.

More specifically, consider the 83-dimensional vector $\boldsymbol{x}$ of EXIF consistency predictions for a pair of patches i and j. We estimate the overall consistency between the patches as $c_{ij} = p_\theta(y \mid \boldsymbol{x})$ where p_θ is a two-layer MLP with 512 hidden units. The network is trained to predict whether i and j come from the same training image (i.e. $y = 1$ if they're the same; $y = 0$ if they're different). This has the effect of calibrating the different EXIF predictions while modeling correlations between them.

3.4 Directly Predicting Image Consistency

One might ask whether a model can learn a measure of consistency by directly predicting whether two image patches come from the same image. The main problem is that

such a model, which is trained on arbitrary image pairs, will be tempted to focus on easy cues because they are sufficient to solve the vast majority of cases. For example, the network might simply learn to compare color histograms, which is a surprisingly powerful cue for same/different image classification task [35, 30]. Of course, given infinite training data and training time, we would expect this model to eventually learn the same cues as the metadata model. To study this further, we also train a Siamese network, similar in structure to the EXIF-consistency model (Section 3.1), to solve the task of same-or-different image consistency.

3.5 From Patch Consistency to Image Self-Consistency

So far we have introduced models that can measure some form of consistency between pairs of patches. In order to transform this into something usable for detecting splices, we need to aggregate these pairwise consistency probabilities into a global self-consistency score for the entire image.

Given an image, we sample patches in a grid, using a stride such that the number of patches sampled along the longest image dimension is 25. This results in at most 625 patches (for the common 4:3 aspect ratio, we sample $25 \times 18 = 450$ patches). For a given patch, we can visualize a response map corresponding to its consistency with every other patch in the image. To increase the spatial resolution of each response map, we average the predictions of overlapping patches. If there is a splice, then the majority of patches from the untampered portion of the image will ideally have low consistency with patches from the tampered region (see Figure 6c).

To produce a single response map for an input image, we want to find the most consistent mode among all patch response maps. We do this mode-seeking using Mean Shift [36]. The resulting response map naturally segments the image into consistent and inconsistent regions (see Figure 6d). We call the merged response map a *consistency map*. We can also qualitatively visualize the tampered image region by clustering the affinity matrix, e.g. with Normalized Cuts [37].

To help understand how different EXIF attributes vary in their consistency predictions, we created response maps for each tag for an example image (Figure 7). While the individual tags provide a noisy consistency signal, the merged response map accurately localizes the spliced region.

4 Results

We evaluate our self-consistency model on two closely related tasks: splice detection and splice localization. In the former, our goal is to simply classify images as being spliced *vs.* authentic. In the latter, the goal is to localize the spliced regions within an image.

4.1 Benchmarks

We evaluate our method on five different datasets. This includes three existing datasets: the widely used *Columbia* dataset [38], which consists of 180 relatively simple splices,

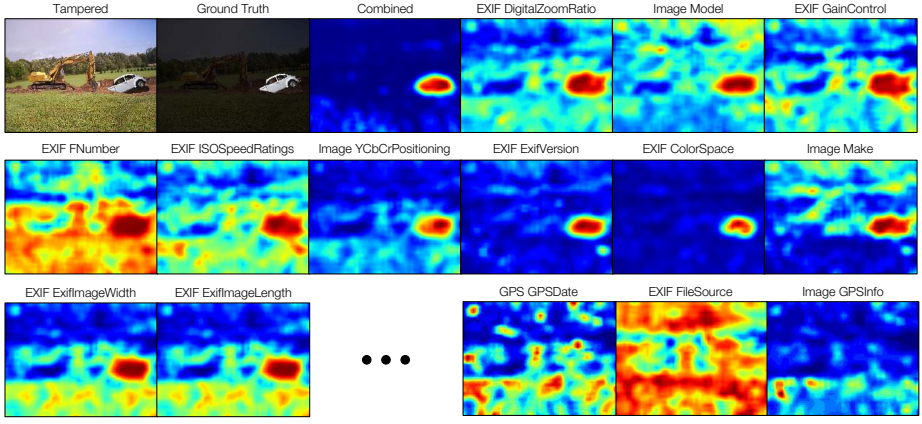

Fig. 7: **Consistency map from different EXIF tags:** We compute consistency maps for each metadata attribute independently (response maps sorted by localization accuracy). The merged consistency map accurately localizes the spliced car.

and two more challenging datasets, *Carvalho* et al. [39] (94 images) and *Realistic Tampering* [40] (220 images), which combine splicing with post-processing operations. The latter also includes other tampering operations, such as copy-move.

One potential shortcoming of these existing datasets is that they were created by a small number of artists and may not be representative of the variety of forgeries encountered online. To address this issue, we introduce a new *In-the-Wild* forensics dataset that consists of 201 images scraped from THE ONION, a parody news website (i.e. fake news), and REDDIT PHOTOSHOP BATTLES, an online community of users who create and share manipulated images (which has been used in other recent forensics work [41]). Since ground truth labels are not available for internet splices, we annotated the images by hand and obtain approximate ground truth (using the unmodified source images as reference when they were available).

Finally, we also want to evaluate our method on automatically-generated splices. For this, we used the scene completion data from Hays and Efros [1], which comes with inpainting results, masks, and source images for a total of 55 images. We note that the ground-truth masks are only approximate, since the scene completion algorithm may alter a small region of pixels outside the mask in order to produce seamless splices.

4.2 Comparisons

We compared three baseline algorithms, all of which use classic image processing techniques to explicitly model and exploit imaging artifacts: Color Filter Array (CFA) [42] detects artifacts in color pattern interpolation; JPEG DCT [43] detects inconsistencies over JPEG coefficients; and Noise Variance (NOI) [44] detects anomalous noise patterns using wavelets. We used implementations of these algorithms provided by [45].

Since we also wanted to compare our unsupervised method with approaches that were trained on labeled data, we report results from a learning-based method: E-MFCN

Dataset	**Columbia [38]**	**Carvalho [39]**	**RT [40]**
CFA [42]	0.83	0.64	0.54
DCT [43]	0.58	0.63	0.52
NOI [44]	0.73	0.66	0.52
Supervised FCN	0.57	0.56	**0.56**
Camera Classification	0.70	0.73	0.15
Image-Consistency	0.96	0.65	0.35
EXIF-Consistency	**0.98**	**0.87**	0.55

Table 1: **Splice Detection:** We compare our splice detection accuracy on 3 different datasets. For each one, we measure the mean average precision (mAP) of detecting whether an image has been spliced. We note that RT is a dataset that contains a variety of manipulations (not just splicing).

[16]. Given a dataset of spliced images and masks as training data, they use a supervised fully convolutional network (FCN) [46] to predict splice masks and boundaries in test images. As a supervised method, their model requires a large training set of splices. To test on our new datasets, we implemented a simplified version of their model (a standard FCN trained to recognize spliced pixels) that was trained with a training split of the *Columbia*, *Carvalho*, and *Realistic Tampering* datasets. We split every dataset in half to construct train/test sets.

Finally, we present two simplified versions of our full *EXIF-Consistency* model. The first, *Camera-Classification*, was trained to directly predict which camera model produced a given image patch. We evaluate the output of the camera classification model by sampling image patches from a test image and assigning the most frequently predicted camera as the natural image and everything else as the spliced region. We consider an image to be untampered when every patch's predicted camera model is the same.

The second model, *Image-Consistency*, is a network that directly predicts whether two patches are sampled from the same image (Section 3.4). An image is considered likely to have been tampered if its constituent patches are predicted to have come from different images. The evaluations of the simplified models are performed the same way as our full *EXIF-Consistency* model.

We trained all our models, including our variant models, using a ResNet50 [34] pretrained on ImageNet [47]. We used a batch size of 128 and optimized our objective using Adam [48] with a learning rate of 10^{-4}. For *EXIF-Consistency* and *Image-Consistency*, we report our results after training for 1 million iterations. The 2-layer MLP used to compute patch consistency on top of the *EXIF-Consistency* model predictions was trained for $10,000$ iterations.

4.3 Splice Detection

We evaluate splice detection using the three datasets that contain both untampered and manipulated images: *Columbia*, *Carvalho*, and *Realistic Tampering*. For each algorithm, we extract the localization map and obtain an overall score by spatially averaging the responses. The images are ranked based on their overall scores, and we compute the mean average precision (mAP) for the whole dataset.

Dataset	**Columbia** [38]	**Carvalho** [39]	**RT** [40]	**In-the-Wild**	**Hays** [1]
CFA [42]	0.84	0.58	**0.69**	0.59	0.58
DCT [43]	0.58	0.60	0.53	0.63	0.52
NOI [44]	0.67	0.64	0.57	0.64	0.59
Supervised FCN	0.80	0.58	0.58	0.59	0.57
Camera Classification	0.59	0.53	0.51	0.53	0.55
Image-Consistency	0.91	0.67	0.58	0.69	0.69
EXIF-Consistency	**0.97**	**0.75**	0.59	**0.72**	**0.75**

Table 2: **Splice Localization:** We evaluate our model on 5 datasets using a mean average precision-based metric (p-mAP).

Dataset	**Columbia** [38]		**Carvalho** [39]	
Metric	MCC	F1	MCC	F1
CFA [42]	0.23	0.47	0.16	0.29
DCT [43]	0.33	0.52	0.19	0.31
NOI [44]	0.41	0.57	0.25	0.34
E-MFCN [16]	0.48	0.61	**0.41**	0.48
Camera Classification	0.29	0.51	0.15	0.55
Image-Consistency	0.58	0.62	0.17	0.71
EXIF-Consistency	**0.70**	**0.72**	0.32	**0.86**

Table 3: **Comparison against Salloum et. al:** We compare against numbers reported by [16] for splice localization. We note that they compute their numbers after finding individual threshold that optimizes their score per image, while we threshold our probabilities at 0.5.

Table 1 shows the mAP for detecting manipulated images. Our *EXIF-Consistency* model achieves state-of-the-art performance on *Columbia* and *Carvalho* while producing results comparable to supervised *FCN* on Realistic Tampering.

4.4 Splice Localization

Having seen that our model can distinguish spliced and authentic images, we next ask whether it can also localize spliced regions within images. For each image in our dataset, our algorithm produces an unnormalized probability that each pixel is part of a splice.

Since it is sometimes ill-defined which of the two segments is spliced (e.g. when the inserted region is approximately half of the image), we evaluate our performance using a metric that is invariant to permutations of the two regions. We use a metric similar to mAP, but with symmetry and permutation invariance built in (we denote it *p-mAP*). Given an image's ground-truth binary mask m and unnormalized splice probabilities p, we compute $\max(s(m,p), s(1-m,p))$, where $s(m,p)$ is symmetrized average precision: $\frac{1}{2}(AP(m,p) + AP(1-m,1-p))$. We also evaluated our results using MCC and F1 measures [2], so that we could directly compare with previous forensics work [16]. These metrics evaluate a binary segmentation and hence require thresholding our predicted probabilities. We threshold our predictions at $p = 0.5$ for all images. Since [16]

[2] F1 score is defined as $\frac{2TP}{2TP+FN+FP}$ and MCC as $\frac{(TP\times TN)-(FP\times FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$.

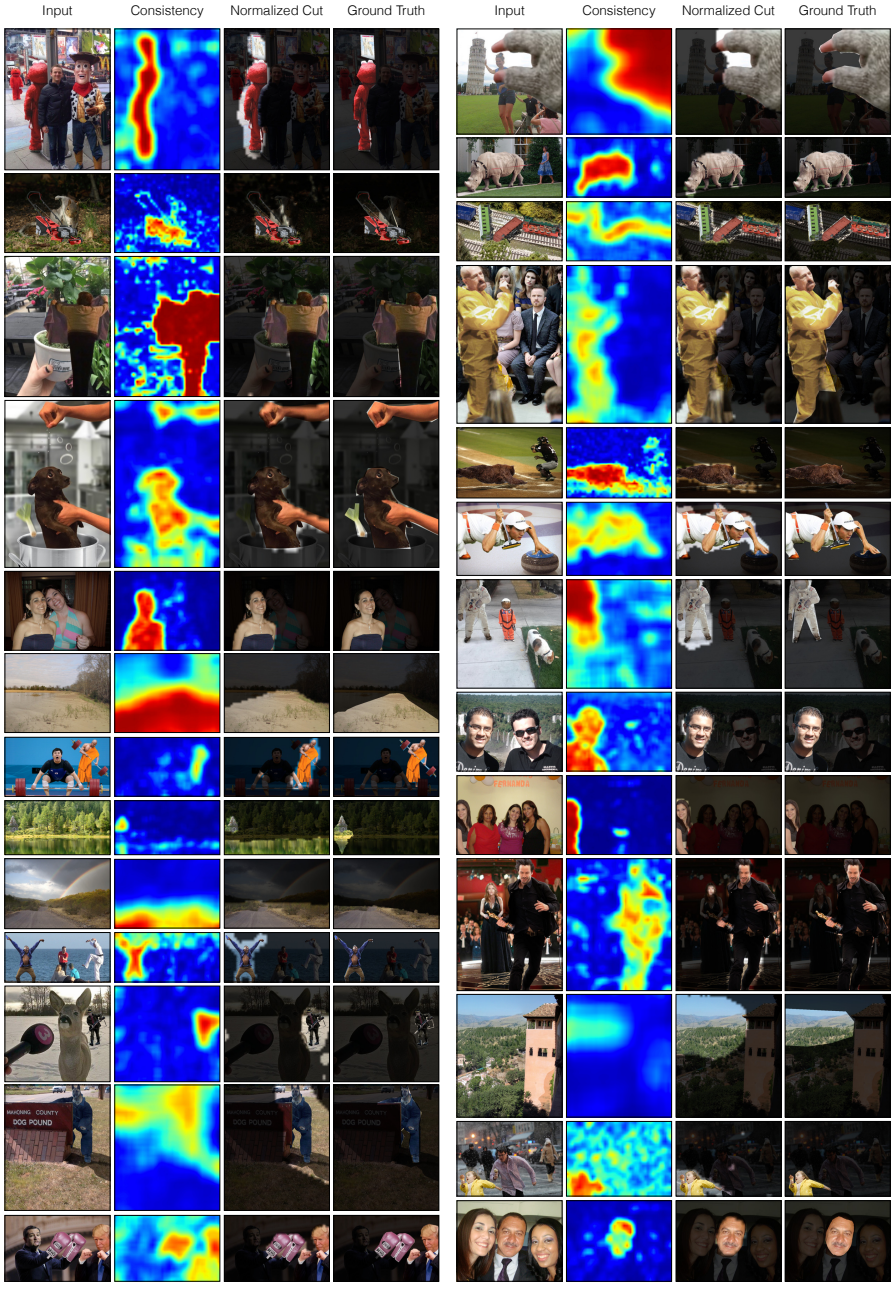

Fig. 8: **Detecting Fakes:** *EXIF Consistency* successfully localizes manipulations across many different datasets. We show qualitative results on images from *Carvalho*, *In-the-Wild*, *Hays* and *Realistic Tampering*.

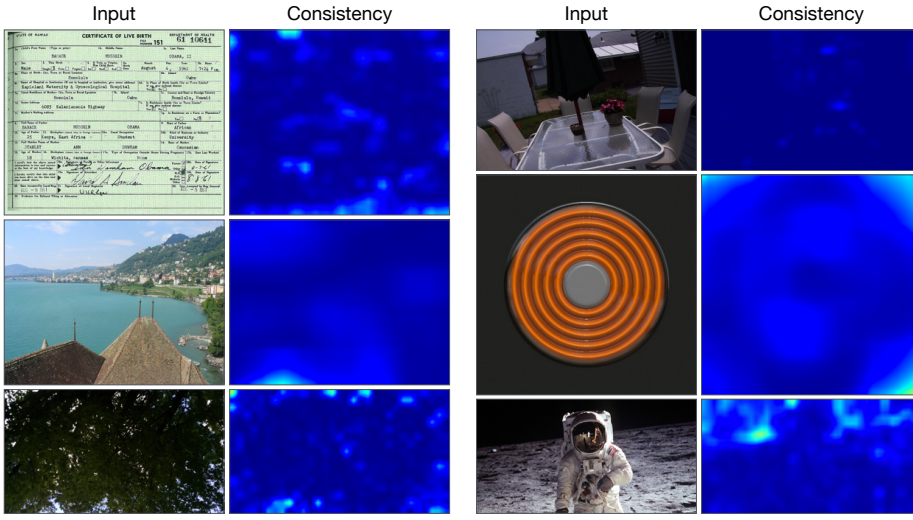

Fig. 9: **Response on Untampered Images:** Our algorithm's response map contains fewer inconsistencies when given an untampered images.

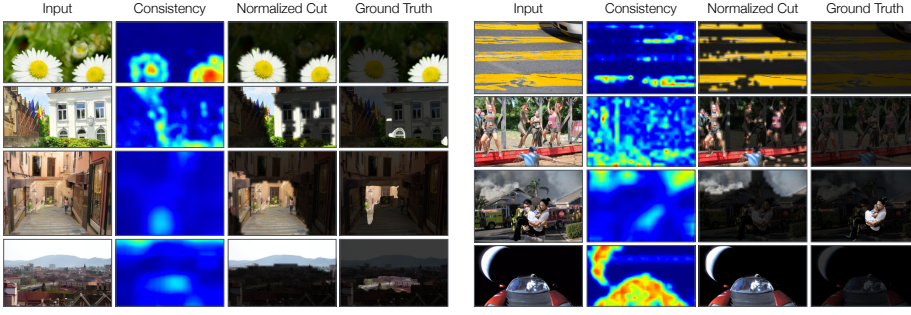

Fig. 10: **Failure Cases:** We present typical failure modes of our model. As we can see with outdoor images, overexposure frequently leads to false positives in the sky. In addition some splices are too small that we cannot effectively locate them using consistency. Finally, the flower example produces a partially incorrect result when using the *EXIF Consistency* model. Since the manipulation was a copy-move, the manipulation is only detectable via post-processing consistency cues (and not EXIF-consistency cues).

reported their numbers on the full *Columbia* and *Carvalho* datasets (rather than our test split), we evaluated our methods on the full dataset and report the comparison in Table 3.

The quantitative results on Table 2 show that our *Self-Consistency* model achieves the best performance across all datasets with the exception of the *Realistic Tampering (RT)* dataset. Notably, the model generally outperformed the supervised baselines, which were trained with actual manipulated images, despite the fact that our model never saw a tampered image during training. We suspect that the supervised models'

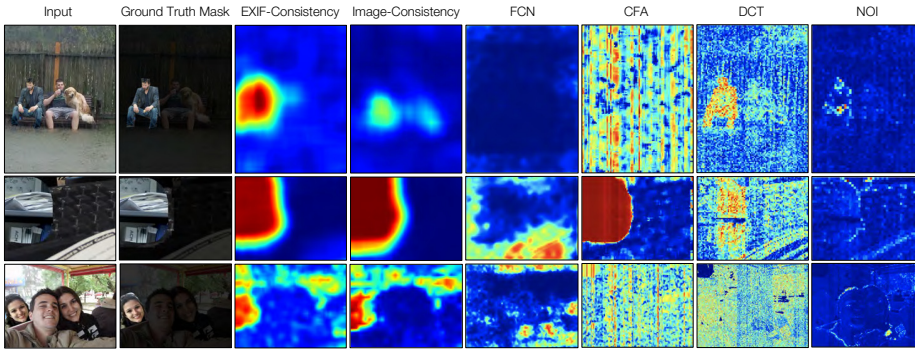

Fig. 11: **Comparing Methods:** We visualize the qualitative difference between *Self-Consistency* and baselines. Our model can correctly localizes image splices from *In-the-Wild*, *Columbia* and *Carvalho* that other methods make mistakes on.

poor performance may be due to the difficulty in generalizing to manipulations made by artists whose works were not present at training time. In Figure 5, we show the model's performance on the Columbia dataset when using individual EXIF attributes (rather than the learned "overall" consistency).

We also observed that *EXIF-Consistency* consistently outperformed *Image-Consistency* across different datasets. We suspect that this may be due to "shortcuts" *Image-Consistency* learns during training (3.4). For example, during training it often suffices to analyze the similarity of the content of the patches (e.g., their color histograms). At test time, however, these shortcuts may be maladaptive since a forger will have intentionally concealed such obvious cues.

It is also instructive to look at the qualitative results of our method, which we show in Figure 8. We see that our method can localize manipulations on a wide range of different splices. Furthermore, in Figure 9, we show that our method produces highly consistent predictions when tested on real images. We can also look at the qualitative differences between our method and the baselines in Figure 11.

Finally, we ask which EXIF tags were useful for performing the splice localization task. To study this, we computed a response map for individual tags on the Columbia dataset, which we show in Figure 7. We see that the most successful tags tend to be associated with the dimensions of the camera and the camera model.

Failure cases. In Figure 10 we show some common failure cases. Our performance on *Realistic Tampering* illustrates some shortcomings with *EXIF-Consistency*. First, our model is not well-suited to finding very small splices, such as the ones that appear in *RT*. When spliced regions are small, the model's large stride may skip over spliced regions, mistakenly suggesting that no manipulations exist. Second, over- and under-exposed regions are sometimes flagged by our model to be inconsistent because they lack any meta-data signal (e.g. because they are nearly uniformly black or white). Finally, *RT* contains a significant number of additional manipulations, such as copy-move, that cannot be consistently detected via meta-data consistency since the manipulated content comes from exactly the same photo.

Running time. In our experiments, computing the consistency-map took approximately 16 seconds per image (e.g. Figure 11). Most of the time goes into computing consistency comparisons between patches. To reduce the running time, we reuse the convolutional features for each patch between comparisons. Note that we could also significantly improve the speed by doing only a subset of the pairwise comparisons, as in affinity-based clustering work [37], or by decreasing the resolution of the response map by increasing patch-sampling stride.

5 Discussion

In this paper, we have proposed a self-supervised method for detecting image manipulations. Our experiments show that the proposed method obtains state-of-the-art results on several datasets, even though it does not use labeled data during training. However, our work also raises a number of questions. In contrast to physically motivated forensics methods [2], our model's results are not easily interpretable, and in particular, it is not clear which visual cues it uses to solve the task. It also remains an open question how best to fuse consistency measurements across an image for localizing manipulations. Finally, while our model is trained without any human annotations, it is still affected in complex ways by design decisions that went into the self-supervision task, such as the ways that EXIF tags were balanced during training.

Self-supervised approaches to visual forensics hold the promise of generalizing to a wide range of manipulations — potentially beyond those that can feasibly be learned through supervised training. However, for a forensics algorithm to be truly general, it must also model the actions of intelligent forgers that adapt to the detection algorithms. Work in adversarial machine learning [49, 50] suggests that having a self-learning forger in the loop will make the forgery detection problem much more difficult to solve, and will require new technical advances.

As new advances in computer vision and image-editing emerge, there is an increasingly urgent need for effective visual forensics methods. We see our approach, which successfully detects manipulations without seeing examples of manipulated images, as being an initial step toward building general-purpose forensics tools.

Acknowledgements

This work was supported, in part, by DARPA MediFor program and UC Berkeley Center for Long-Term Cybersecurity. We thank Hany Farid and Shruti Agarwal for their advice, assistance, and inspiration in building this project, David Fouhey and Saurabh Gupta and Allan Jabri for helping with the editing, Peng Zhou for helping with experiments, and Abhinav Gupta for letting us use his GPUs. Finally, we thank the many *Reddit* and *Onion* artists who unknowingly contributed to our dataset.

References

1. Hays, J., Efros, A.A.: Scene completion using millions of photographs. In: ACM Transactions on Graphics (TOG). Volume 26., ACM (2007) 4 1, 2, 9, 13

2. Farid, H.: Photo forensics. MIT Press (2016) 2, 3, 15
3. Zhu, J.Y., Krahenbuhl, P., Shechtman, E., Efros, A.A.: Learning a discriminative model for the perception of realism in composite images. In: The IEEE International Conference on Computer Vision (ICCV). (December 2015) 2
4. Tsai, Y.H., Shen, X., Lin, Z., Sunkavalli, K., Lu, X., Yang, M.H.: Deep image harmonization. In: CVPR. (2017) 2
5. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Trans. Graph. **28**(3) (2009) 24–1 2
6. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2016) 2
7. Suwajanakorn, S., Seitz, S.M., Kemelmacher-Shlizerman, I.: Synthesizing obama: learning lip sync from audio. ACM Transactions on Graphics (TOG) **36**(4) (2017) 95 2
8. Chung, J.S., Jamaludin, A., Zisserman, A.: You said that? arXiv preprint arXiv:1705.02966 (2017) 2
9. of Standards, N.I., Technology: The 2017 nimble challenge evaluation datasets. https://www.nist.gov/itl/iad/mig/nimble-challenge 2
10. Liu, Q.: Detection of misaligned cropping and recompression with the same quantization matrix and relevant forgery. (2011) 3
11. Luo, W., Huang, J., Qiu, G.: Jpeg error analysis and its applications to digital image forensics. IEEE Transactions on Information Forensics and Security **5**(3) (2010) 480–491 3
12. Huang, F., Huang, J., Shi, Y.Q.: Detecting double jpeg compression with the same quantization matrix. IEEE Transactions on Information Forensics and Security **5**(4) (2010) 848–856 3
13. Popescu, A.C., Farid, H.: Exposing digital forgeries by detecting traces of resampling. IEEE Transactions on signal processing **53**(2) (2005) 758–767 3
14. Swaminathan, A., Wu, M., Liu, K.R.: Digital image forensics via intrinsic fingerprints. **3**(1) (2008) 101–117 3
15. Agarwal, S., Farid, H.: Photo forensics from jpeg dimples. Workshop on Image Forensics and Security (2017) 3
16. Salloum, R., Ren, Y., Kuo, C.J.: Image splicing localization using A multi-task fully convolutional network (MFCN). CoRR **abs/1709.02016** (2017) 3, 9, 13, 14
17. Barni, M., Bondi, L., Bonettini, N., Bestagini, P., Costanzo, A., Maggini, M., Tondi, B., Tubaro, S.: Aligned and non-aligned double JPEG detection using convolutional neural networks. CoRR **abs/1708.00930** (2017) 3
18. Amerini, I., Uricchio, T., Ballan, L., Caldelli, R.: Localization of jpeg double compression through multi-domain convolutional neural networks. In: Proc. of IEEE CVPR Workshop on Media Forensics. (2017) 3
19. Wen, L., Qi, H., Lyu, S.: Contrast enhancement estimation for digital image forensics. arXiv preprint arXiv:1706.03875 (2017) 3
20. Bondi, L., Baroffio, L., Güera, D., Bestagini, P., Delp, E.J., Tubaro, S.: First steps toward camera model identification with convolutional neural networks. IEEE Signal Processing Letters **24**(3) (March 2017) 259–263 3, 5
21. Bondi, L., Lameri, S., Güera, D., Bestagini, P., Delp, E.J., Tubaro, S.: Tampering detection and localization through clustering of camera-based cnn features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. (2017) 1855–1864 3
22. Owen Mayer, M.C.S.: Learned forensic source similarity for unknown camra models. IEEE International Conference on Acoustics, Speech and Signal Processing (2018) 3

23. Chen, B.C., Ghosh, P., Morariu, V.I., Davis., L.S.: Detection of metadata tampering through discrepancy between image content and metadata using multi-task deep learning. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2017) 4
24. de Sa, V.: Learning classification with unlabeled data. In: Neural Information Processing Systems. (1994) 4
25. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. ICCV (2015) 4, 5
26. Jayaraman, D., Grauman, K.: Learning image representations tied to ego-motion. In: ICCV. (December 2015) 4
27. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV. (2015) 4
28. Owens, A., Wu, J., McDermott, J.H., Freeman, W.T., Torralba, A.: Ambient sound provides supervision for visual learning. (2016) 4
29. Zhang, R., Isola, P., Efros, A.A.: Split-brain autoencoders: Unsupervised learning by cross-channel prediction. (2017) 4
30. Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. (2016) 4, 8
31. Kuthirummal, S., Agarwala, A., Goldman, D.B., Nayar, S.K.: Priors for large photo collections and what they reveal about cameras. In: European conference on computer vision, Springer (2008) 74–87 4
32. Hoai, M., De la Torre, F.: Max-margin early event detectors. International Journal of Computer Vision **107**(2) (2014) 191–202 4
33. Mahadevan, V., Li, W., Bhalodia, V., Vasconcelos, N.: Anomaly detection in crowded scenes. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE (2010) 1975–1981 4
34. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778 5, 10
35. Lalonde, J.F., Efros, A.A.: Using color compatibility for assessing image realism. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE (2007) 1–8 8
36. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(8) (Aug 1995) 790–799 8
37. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(8) (Aug 2000) 888–905 8, 15
38. Ng, T.T., Chang, S.F.: A data set of authentic and spliced image blocks. (2004) 9, 10, 13
39. d. Carvalho, T.J., Riess, C., Angelopoulou, E., Pedrini, H., d. R. Rocha, A.: Exposing digital image forgeries by illumination color classification. IEEE Transactions on Information Forensics and Security **8**(7) (July 2013) 1182–1194 9, 10, 13
40. Korus, P., Huang, J.: Evaluation of random field models in multi-modal unsupervised tampering localization. In: Proc. of IEEE Int. Workshop on Inf. Forensics and Security. (2016) 9, 10, 13
41. Moreira, D., Bharati, A., Brogan, J., Pinto, A., Parowski, M., Bowyer, K.W., Flynn, P.J., Rocha, A., Scheirer, W.J.: Image provenance analysis at scale. arXiv preprint arXiv:1801.06510 (2018) 9
42. Ferrara, P., Bianchi, T., Rosa, A.D., Piva, A.: Image forgery localization via fine-grained analysis of cfa artifacts. IEEE Trans. Information Forensics and Security **7**(5) (2012) 1566–1577 9, 10, 13
43. Ye, S., Sun, Q., Chang, E.C.: Detecting digital image forgeries by measuring inconsistencies of blocking artifact. In: ICME07. (2017) 9, 10, 13
44. Mahdian, B., Saic, S.: Using noise inconsistencies for blind image forensics. In: IVC09. (2009) 9, 10, 13

45. Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y., Bouwmeester, R., Spangenberg, J.: Web and social media image forensics for news professionals. In: Social Media In the Newsroom, SMNews16@CWSM, Tenth International AAAI Conference on Web and Social Media workshops. (2016) 9
46. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. CoRR **abs/1605.06211** (2016) 9
47. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR09. (2009) 10
48. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014) 10
49. Ian J. Goodfellow, Y.B.: Generative adversarial networks. arXiv preprint arXiv:1406.2661 (2014) 15
50. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013) 15

A1 Appendix

EXIF attribute definitions We have abbreviated the definitions that were originally sourced from http://www.exiv2.org/tags.html. Please visit our website for additional EXIF information such as: distributions, common values, and prediction rankings.

EXIF Attribute	Definition
EXIF BrightnessValue	The value of brightness.
EXIF ColorSpace	The color space information tag is always recorded as the color space specifier. Normally sRGB is used to define the color space based on the PC monitor conditions and environment. If a color space other than sRGB is used, Uncalibrated is set. Image data recorded as Uncalibrated can be treated as sRGB when it is converted to FlashPix.
EXIF ComponentsConfiguration	Information specific to compressed data. The channels of each component are arranged in order from the 1st component to the 4th. For uncompressed data the data arrangement is given in the tag. However, since can only express the order of Y, Cb and Cr, this tag is provided for cases when compressed data uses components other than Y, Cb, and Cr and to enable support of other sequences.
EXIF CompressedBitsPerPixel	Specific to compressed data; states the compressed bits per pixel.
EXIF Contrast	This tag indicates the direction of contrast processing applied by the camera when the image was shot.
EXIF CustomRendered	This tag indicates the use of special processing on image data, such as rendering geared to output. When special processing is performed, the reader is expected to disable or minimize any further processing.
EXIF DateTimeDigitized	The date and time when the image was stored as digital data.
EXIF DateTimeOriginal	The date and time when the original image data was generated.
EXIF DigitalZoomRatio	This tag indicates the digital zoom ratio when the image was shot. If the numerator of the recorded value is 0, this indicates that digital zoom was not used.
EXIF ExifImageLength	The number of rows of image data. In JPEG compressed data a JPEG marker is used instead of this tag.
EXIF ExifImageWidth	The number of columns of image data, equal to the number of pixels per row. In JPEG compressed data a JPEG marker is used instead of this tag.
EXIF ExifVersion	The version of this standard supported. Nonexistence of this field is taken to mean nonconformance to the standard
EXIF ExposureBiasValue	The exposure bias.
EXIF ExposureMode	This tag indicates the exposure mode set when the image was shot. In auto-bracketing mode, the camera shoots a series of frames of the same scene at different exposure settings.
EXIF ExposureProgram	The class of the program used by the camera to set exposure when the picture is taken.
EXIF ExposureTime	Exposure time, given in seconds.
EXIF FileSource	Indicates the image source. If a DSC recorded the image, this tag value of this tag always be set to 3, indicating that the image was recorded on a DSC.
EXIF Flash	Indicates the status of flash when the image was shot.
EXIF FlashPixVersion	The FlashPix format version supported by a FPXR file.
EXIF FNumber	The F number.
EXIF FocalLength	The actual focal length of the lens, in mm.
EXIF FocalLengthIn35mmFilm	This tag indicates the equivalent focal length assuming a 35mm film camera, in mm. A value of 0 means the focal length is unknown. Note that this tag differs from the tag.
EXIF FocalPlaneResolutionUnit	Unit of measurement for FocalPlaneXResolution and FocalPlaneYResolution.
EXIF FocalPlaneXResolution	Number of pixels per FocalPlaneResolutionUnit in ImageWidth direction for main image.
EXIF FocalPlaneYResolution	Number of pixels per FocalPlaneResolutionUnit in ImageLength direction for main image.
EXIF GainControl	This tag indicates the degree of overall image gain adjustment.
EXIF InteroperabilityOffset	Unknown
EXIF ISOSpeedRatings	Indicates the ISO Speed and ISO Latitude of the camera or input device as specified in ISO 12232.
EXIF LensMake	This tag records the lens manufacturer as an ASCII string.
EXIF LensModel	This tag records the lens's model name and model number as an ASCII string.
EXIF LensSpecification	This tag notes minimum focal length, maximum focal length, minimum F number in the minimum focal length, and minimum F number in the maximum focal length, which are specification information for the lens that was used in photography. When the minimum F number is unknown, the notation is 0/0
EXIF LightSource	The kind of light source.

EXIF MaxApertureValue	The smallest F number of the lens.
EXIF MeteringMode	The metering mode.
EXIF OffsetSchema	Unknown
EXIF Saturation	This tag indicates the direction of saturation processing applied by the camera when the image was shot.
EXIF SceneCaptureType	This tag indicates the type of scene that was shot. It can also be used to record the mode in which the image was shot. Note that this differs from the tag.
EXIF SceneType	Indicates the type of scene. If a DSC recorded the image, this tag value must always be set to 1, indicating that the image was directly photographed.
EXIF SensingMethod	Type of image sensor.
EXIF SensitivityType	The SensitivityType tag indicates which one of the parameters of ISO12232 is the PhotographicSensitivity tag.
EXIF Sharpness	This tag indicates the direction of sharpness processing applied by the camera when the image was shot.
EXIF ShutterSpeedValue	Shutter speed.
EXIF SubjectArea	This tag indicates the location and area of the main subject in the overall scene.
EXIF SubjectDistanceRange	This tag indicates the distance to the subject.
EXIF SubSecTime	A tag used to record fractions of seconds for the tag.
EXIF SubSecTimeDigitized	A tag used to record fractions of seconds for the tag.
EXIF SubSecTimeOriginal	A tag used to record fractions of seconds for the tag.
EXIF UserComment	A tag for Exif users to write keywords or comments.
EXIF WhiteBalance	This tag indicates the white balance mode set when the image was shot.
GPS GPSAltitude	Indicates the altitude based on the reference in GPSAltitudeRef.
GPS GPSAltitudeRef	Indicates the altitude used as the reference altitude.
GPS GPSDate	A character string recording date and time information relative to UTC (Coordinated Universal Time).
GPS GPSImgDirection	Indicates the direction of the image when it was captured.
GPS GPSImgDirectionRef	Indicates the reference for giving the direction of the image when it is captured.
GPS GPSLatitude	Indicates the latitude.
GPS GPSLatitudeRef	Indicates whether the latitude is north or south latitude.
GPS GPSLongitude	Indicates the longitude.
GPS GPSLongitudeRef	Indicates whether the longitude is east or west longitude.
GPS GPSTimeStamp	Indicates the time as UTC (Coordinated Universal Time).
GPS GPSVersionID	Indicates the version of GPS.
Image Artist	This tag records the name of the camera owner, photographer or image creator.
Image Copyright	Copyright information.
Image ExifOffset	Image ExifOffset.
Image GPSInfo	A pointer to the GPS Info IFD.
Image ImageDescription	A character string giving the title of the image.
Image Make	The manufacturer of the recording equipment.
Image Model	The model name or model number of the equipment.
Image Orientation	The image orientation viewed in terms of rows and columns.
Image PrintImageMatching	Print Image Matching, description needed.
Image ResolutionUnit	The unit for measuring YResolution and XResolution. The same unit is used for both.
Image Software	This tag records the name and version of the software or firmware of the camera or image input device used to generate the image.
Image XResolution	Number of pixels per FocalPlaneResolutionUnit in ImageWidth direction for main image.
Image YCbCrPositioning	The position of chrominance components in relation to the luminance component.
Image YResolution	Number of pixels per FocalPlaneResolutionUnit in ImageLength direction for main image.
Inter InteroperabilityIndex	Indicates the identification of the Interoperability rule.
Inter InteroperabilityVersion	Interoperability version.
Inter RelatedImageLength	Image height.
Inter RelatedImageWidth	Image width.