Towards Open-Set Forensic Source Grouping on JPEG Header Information

Patrick Mullan, Christian Riess, Felix Freiling

IT Security Infrastructures Lab
Friedrich-Alexander University Erlangen-Nürnberg

Abstract

Image provenance, i.e., information on the model and make of the device that was used to produce an image, is a helpful cue in many digital investigations. Such information can, for example, help refute the hypothesis that an illegal photograph found on the Internet was produced using the personal device of a suspect. Grouping images by provenance can be done in different ways. Based on the encouraging insights from previous works, we consider the grouping of JPEG images by their file headers where previous work performed image classification on a closed-set of devices. However, due to the ongoing development of new camera models and the practical difficulty to keep a model database up-to-date, we propose to treat image provenance as an open-set classification problem with the goal to predict the make of a previously unseen camera model. We show that such a prediction can performe remarkably well, with median accuracies beyond 90% for each make. In a more challenging experiment with images that were postprocessed, we achieve a median accuracy of about 55% and 75% for desktop software and for smartphone apps, respectively.

Keywords: Image Forensics, JPEG Metadata, Headerdata, Blind Source Separation

1. Introduction

Off-the-shelf digital hard drives can nowadays store terabytes of data. Oftentimes, a considerable portion of this space is used for multimedia content, such as images and videos. In a forensic investigation, it is not uncommon to examine hard drives with thousands of images. An investigator is forced to sift through this data if some of the images on the hard drive are assumed to be connected to the case. Investigators therefore need methods to quickly and reliably group these images into helpful classes. Next to an assessment of image content, a hint on the device with which the image was taken can be an important lead in an investigation because hypotheses regarding the connection between a particular image found on the web and a perpetrator's device can be substantiated or refuted.

In the literature, several methods for determining image provenance can be potentially useful for a forensic analyst to sort the images on a hard drive into its various sources. The largest part of this body of work analyzes specific pixel statistics in the image that allow to associate it with a camera or a specific processing chain. This approach has many advantages, most notably the exceedingly high reliability of some pixel-level traces like the photo-response non-uniformity (PRNU) (Lukás et al., 2006). The core idea in PRNU is, that each camera sensor, has a unique pattern of unavoidable imperfections, introduced at the manufacturing process. These device identifying traces can be averaged out across a sufficient amount of images from the device with sophisticated signal processing algorithms. Therefore, this approach also come at a cost, with respect to required training data, and with the computational effort to analyze a large number of images at pixel level. The approach does an analysis on device level, and not model or make level which also can restrict its applicability in real world scenarios.

A much cheaper approach to determining image provenance is to sort images based on their header information. Since it has often been claimed that header information is easy to manipulate and therefore not very reliable, this topic has largely been overlooked in the multimedia forensics community, with only few works on images (e.g., Kee et al. (2011); Gloe (2012); Mullan et al. (2019)) and on video (e.g., Iuliani et al. (2019); Gloe et al. (2014); Güera et al. (2019)). Header information has previously been shown to be highly discriminative on JPEG images of consumer cameras (Kee et al., 2011), and some efforts have been made towards images from smartphones (Mullan et al., 2019). Since these methods do not decode the whole image, but instead only read out the header information, they are extremely light-weight, and can be used to rapidly sort coarsely huge amounts of files by camera model or make.

However, one practical issue with such header-based approaches is the maintenance of an accurate database about existing models, since current methods require to know a model in advance in order to correctly sort the images. To maintain a clean model database aims at an extremely fast moving target: every month, new smartphones, compact cameras and DSLRs are presented to the public. Additionally, header information might be set by smartphone apps, which form an even faster changing, virtually unbounded search space for acquisition sources. Camera vendors typically write the camera model and make in the EXIF data. In this case, a database can in principle be updated during operation whenever a previously unseen camera model is found. However, such an automated approach may not be very reliable: we observed on our database of images collected from the Internet that information on model and make is oftentimes not present, either because it has been overwritten by processing software or it has been modified in some other way. Hence, we hypothesize that it is close to impossible to maintain an up-to-date model database with reasonable resources, that does not permanently classify new models as outliers.

As a consequence, we propose to consider the task of forensic source grouping from image headers as an *open-set* problem, and to develop strategies to perform source grouping in presence of an incomplete dataset. We define the open set problem as associating a previously unseen model to a camera make. To our knowledge, this question has not been addressed before in the literature. In this work, we make first contributions in this direction.

The study is performed on the widely used JPEG format. We collected a realistic, only partially controlled dataset, from the photo sharing web site Flickr. On this dataset, we present three investigations:

1. A characterization of the variability of header information on DLSR and compact cameras of two major manufacturers.
2. An experimental evaluation how well the make of a camera can be predicted on images from models that were unseen during training.
3. An experimental evaluation how well the make of a camera can be predicted on images that were processed by software other than the original camera software. This software was also unseen during training.

The source grouping is performed with a random forest classifier on JPEG metadata and JPEG quantization matrices. Our results show that new models in many cases share common parameters with known models. They also show that smartphone apps oftentimes do not significantly change the header information, while desktop software oftentimes does. We hope that these contributions foster further research into more sophisticated machine learning models for open-set header grouping.

The paper is organized as follows. In Sec. 2, we review related work on image provenance. The proposed method, the features and the data collection are presented in Sec. 3. Sec. 4 investigates the variability of header information. We start with an empirical study to show intra-make and inter-make differences. This is followed by experiments to predict makes from unknown models. Subsequently we complete the methodology section with a study on classification of makes from images that were postprocessed, either via apps or via desktop-based software. Finally, in Sec. 5 we conclude.

2. Related Literature

Various aspects for the detection of image provenance have been investigated. A specific subtask is source identification, where the goal is to detect the acquisition device, make, or model of a picture. For example, McKay et al. (2008) propose an early learning-based method to distinguish camera images, scanner images, and computer generated images, and also to predict make and model of the acquisition device. This method uses a Support Vector Machine on handcrafted features like noise statistics and distributions of color coefficients.

The classification of a camera model has also been studied in several other works that use neural network classifiers, for example Bayar and Stamm (2017a,b, 2018); Bondi et al. (2017); Júnior et al. (2019). This task was also subject of the "Forensic Camera Model Identification Challenge" at the IEEE Signal Processing Cup 2018 (Stamm et al., 2018).

Another family of methods for source identification targets at identifying even a specific device. The most successful approach to linking an image to a unique capturing device uses Photo-Response Non-Uniformity (PRNU), reported in Lukás et al. (2006). This approach leverages a sensor property, namely that each camera sensor has unique manufacturing imperfections. These imperfections lead in each pixel to a subtle systematic deviation in the intensity response. A reference fingerprint for a camera can be calculated from a reasonably large set of source images, and then be correlated to an unknown target image. This method is highly accurate, as reported in many works, e.g., Goljan et al. (2009). However, the need for a reference fingerprint also limits its applicability in some practical tasks, since the fingerprint computation requires a sufficiently large amount of training images.

Despite impressive results, pixel-based methods are not free of challenges. Multiple image compressions and downsampling are notorious failure cases that easily destroy the subtle pixel-level statistics. Also, higher-level abstractions of pixel-based association, such as blind source clustering, are computationally extremely costly (Marra et al., 2017). Furthermore, modern smartphones oftentimes produce computational images. Such computational images subject the sensor data to strong processing. For example, Wronski et al. (2019) present Google's approach to computational imaging, where the pixel response is only very loosely related to the actual sensor reading. The impact of these computations on sensor fingerprinting is to our knowledge largely unexplored in the domain of source identification.

Header information may also be impacted by additional processing, and social networks like facebook almost completely replace any existing header information. Nevertheless, many software packages preserve some or all of the header information, and oftentimes leave additional characteristic traces that allow to fingerprint software packages. For example, the work by Gloe (2012) investigates the structure of the containers in a JPEG file to reveal characteristic traces. Gloe et al. (2013) show that also PNG files can expose characteristic meta information. Mullan et al. (2019) show on images from Apple smartphones that characteristic header changes after software updates can be used to fingerprint software versions. Classification of camera models is very lightweight, and can easily be scaled to a large number of camera models. It is, however, somewhat less specific than pixel-based cues for model identification. Kee et al. (2011) investigate how unique headers are to a specific model of DSLR and compact cameras. This work reports that 69.1% of the collected image headers are unique to one specific model. 12.8% of headers are shared between two models. In the worst cases, 14 camera models share identical headers.

The open set problem for source identification received only little attention. To our knowledge, there exist only a few pixel-based methods that introduce a rejection class for unknown models, e.g., Bondi et al. (2017); Bayar and Stamm (2018); Júnior et al. (2019). We believe that a specific reformulation of the open set problem can make it feasible on header data: we consider source identification as a hierarchical task, where one make offers many models on the market. Since it is unlikely that the software stack is fundamentally different from other models of that make, we hypothesize that the make can be successfully predicted although the model itself is unknown.

3. Tackling the Open-Set Problem: The Make-Model-Device Hierarchy

The open-set problem poses the question how an image from a previously unseen camera model can be associated with known training data. Considering header data, it may be the case that the model and make are simply part of the EXIF data. However, this information can be unreliable, as these identifying strings appear in a surprisingly rich multitude of different spelling and capitalization variants, or may be missing or overwritten altogether by processing software. Hence, the goal of this work is to associate header data under the assumption that the EXIF information on make and model is unreliable, and thus we treat camera association as an open-set problem. To tackle the challenge of different string variants, we apply a normalization to unify makes, models and software versions.

To make this open-set problem feasible, we propose to consider the camera model as one element in a hierarchy of identifying camera properties. A previously unseen camera model may still resemble some similarities with cameras of the same make, assuming that the manufacturer reuses some software and/or hardware components across models. Hence, although the make is a less precise cue than the model, we hypothesize that such a prediction is feasible.

Following Mullan et al. (2019), a more formalized presentation of the hierarchy of identifying camera properties is shown in Fig. 1. It consists of an upper and a lower part. The upper part shows different granularity scales for camera association on hardware level, the lower part on software level.

In this hierarchy, hardware granularity ranges from the type of image, e.g., a camera photograph versus a digital scan over the manufacturer like Apple and the model, such as an iPhone, to a particular device. Existing pixel-based methods typically address the two most specific cases, either to identify specific devices such as PRNU, like Lukás et al. (2006); Goljan et al. (2009), or camera models, e.g., Bayar and Stamm (2017a, 2018); Bondi et al. (2017). Software granularity is mostly important with smartphones or other Internet-connected devices. Here, the operating system typically provides the API for capturing and encoding images, but individual photo capturing or management apps may decide not to apply specific individual processing on the acquired images. The main difference to the hardware stack is that software can change through updates, which may also manifest in the processing and encoding of images.

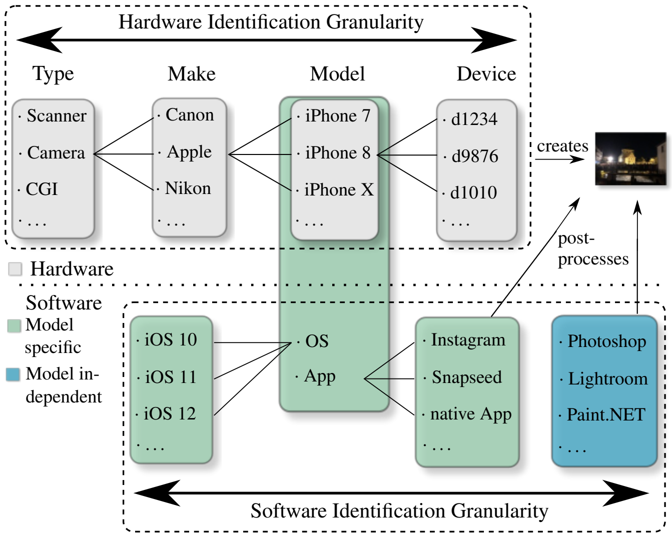


Figure 1: Identification granularity along hardware and software (Mullan et al., 2019). Expanded for potential post-processing, also after the image has been loaded from the device, for example to a desktop machine, running Photoshop.

In this work, we investigate how well previously unseen models can be associated to its make, using image header information. This study performs in three steps. First, we show examples for the variability of header information. Second, we show how well unseen models can be associated to a make. Third, we again consider association of unseen models to make, but this time with the additional difficulty that the image underwent processing from PC software and apps.

The methods for these experiments are presented in the remainder of this Section. The data acquisition is presented in Sec. 3.1. The proposed feature set is presented in Sec. 3.2, and the used classifier in Sec. 3.3.

3.1. Data Acquistion and Preparation

We follow reported practices in the literature for collecting header information (e.g. Kee et al. (2011); Mullan et al. (2019)). In detail, we collected 2,833,349 images and their associated user names from the website Flickr. The images were uploaded between 2008 and 2019. All images that are not in valid JPEG format are discarded. In contrast to previous work, we did not discard images with header information that indicates the use of editing or viewing software, since such common modifications are part of the open set problem in this work.

Header information is directly extracted from an image. The make in the `EXIF:Make` tag is manually normalized. For example, entries like "SAMSUNG", "Samsung techwin co., ltd", and "SAMSUNG TECHWIN" are unified to "Samsung" via regular expressions. These entries are used as ground truth in our experiments. Also models in the `EXIF:Model` tag are manually normalized wherever there are duplicates known to us. For example, the Canon model "EOS 800D" is also distributed as "EOS Rebel T7i" and "EOS Kiss X9i". We remove models and makes with empty strings, evidently malformed names ("cAnon"), and contradictory entries (e.g., model "iPhone" from

Canon

value at key "Exif:Software"	Frequency
NaN	286173
Digital Photo Professional	14772
Picasa	11250
Adobe Photoshop CS3 Windows	9030
Adobe Photoshop CS5 Windows	8682

Nikon

value in key "Exif:Software"	Frequency
Ver.1.00	60633
Ver.1.01	33435
Ver.1.10	11057
Ver.1.02	9306
Ver.1.03	7580

Table 1: Most common EXIF:Software tags for Canon (top) and Nikon (bot.).

make "Canon"). These entries are used in our experiment to split the data into camera models.

The treatment of the `EXIF:Software` tag is slightly more complicated. Some camera manufacturers like Nikon use this field to indicate their firmware version. Also, many packages for image processing like Photoshop write their name into the field `EXIF:Software`. For example, Tab. 1 shows the most frequent entries in `EXIF:Software` for Canon images on top and Nikon images on the bottom. In most cases, Canon cameras do not set the field `EXIF:Software`, indicated by `NaN` in the table. For Nikon, the most frequent values are the firmware version strings. In our data preparation, we are considering commonly occurring camera labels as "unedited" images, and labels from image processing packages as "edited". To this end, image processing tags are also normalized down to the major version we found, e.g., "Adobe Photoshop CS6 (Windows)" and "Adobe Photoshop CS6 (Macintosh)" are considered identical.

Please note that even after careful data cleansing, this publicly sourced data almost inevitably still contains erroneous entries. Hence, the reported results are likely a lower bound for results on a perfectly clean, fully controlled dataset. Nevertheless, we consider this form of data collection as a use case that is close to practical conditions. We provide scripts that allow the download of the dataset from their respective URLs at `https://faui1-gitlab.cs.fau.de/mullanptr/flickr_data`.

3.2. Studied Feature Set

The JPEG compression standard defines how to compress the image to the final byte stream (Wallace, 1992; Pennebaker and Mitchell, 1992). This byte stream needs to be put in a file container. The "JPEG Interchange File Format" (JFIF) (Hamilton, 1992) is the commonly used file container format for JPEG. It specifies additional details on the image and restricts the modes in which an image can be encoded. With respect to the metadata, the "Exchangable Image File Format" (EXIF) was introduced shortly after JFIF. It specifies many optional metadata entries that are compliant to the TIFF standard. These standards leave considerable freedom in the choice of parameters. This enables camera manufacturers and software packages to trade-off file size and image quality and to embed additional information in form of metadata. Conversely, this freedom enables to extract signatures from JPEG headers to identify the device or software in which the image was created.

In this work, we use two groups of header parameters as features. First, metadata parameters that provide further information on the acquisition conditions and the color-profile metadata. Second, parameters for the encoding of the actual image.

The metadata-specific features are derived from the EXIF and color-profile metadata. These entries are key-value pairs that can be directly read out from the header, e.g., using the command line tool `exiftool`.

EXIF groups information in so called "Image File Directories" (IFDs). The first IFD, `IFD0`, records parameters that cover properties of the actual image, e.g., the numbers of rows and columns of the image. The second IFD, `IFD1`, provides the parameters of the thumbnail image if such a thumbnail is embedded. Further commonly occurring IFD directories are `ExifIFD`, `MakerNotes`, and `GPS`. `ExifIFD` is a directory that only contains additional descriptions on the image, such as shutter speed, aperture size, or the date and time of the image acquisition. `MakerNotes` may contain additional information that is embedded from the manufacturer of the device. The directory `GPS` can reveal information about the geolocation where the image was captured. Additionally to these EXIF entries, the `ICCProfile` denotes another group of entries that is defined by the International Color Consortium. The goal of this specification is to provide a standard that ensures colors in visual information are displayed equally, independent of the output device or manufacturer of that device.

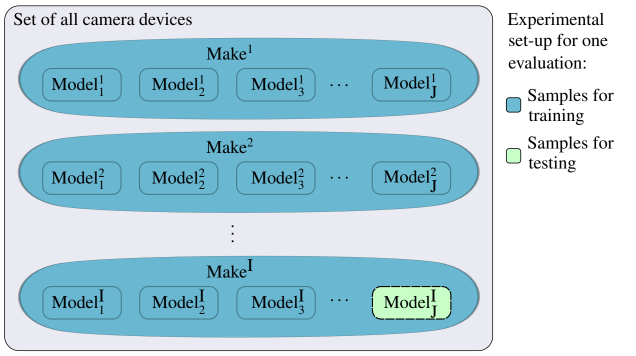
We do not use the entries in the metadata directly, but instead form a histogram of the number of key-value pairs per directory. More specifically, we count the number of entries in each of the five above-stated EXIF directories and the additional `ICCProfile` directory, which leads to a six-dimensional feature vector of integer numbers.

For the encoding-specific parameters, we consider the JPEG quantization tables. We also experimented with the Hufmann tables for runlength-encoding as features (Kee et al., 2011), but did not find them very discriminative on our data.

The JPEG quantization table encodes the coarseness with which individual frequency components of the image are attenuated. As such, it is the key factor in the trade-off between image quality and image size. Manufacturers and software packages hence oftentimes use individual quantization tables (Kee et al., 2011). These tables can be directly read out from the header without decoding the actual image. The quantization matrix is linearized following the JPEG zig-zag pattern, and each coefficient is used as a single feature. In one variant of the evaluation, we only use the quantization matrix for luminosity, which yields a 64-dimensional feature vector. In another variant, we additionally use the chroma matrix, which yields in total a 128-dimensional feature vector.

In one experiment, we combine metadata parameters and quantization parameters for the luminosity matrix, which yields a 70-dimensional feature vector.

3.3. Supervised Learning in the Context of Open-Set Camera Association



Set of all camera devices
Make¹
Model¹₁ Model¹₂ Model¹₃ ··· Model¹_J
Make²
Model²₁ Model²₂ Model²₃ ··· Model²_J
Make^I
Model^I₁ Model^I₂ Model^I₃ ··· Model^I_J
Experimental set-up for one evaluation:
Samples for training
Samples for testing

Figure 2: Relation between makes and models – Models are subsets of makes, and makes form disjoint sets. In our classification experiments, one model (shaded green) is left out of the training data. All other models (shaded blue) are used to train the classifier. Classes are the makes.

In supervised learning, a classifier is fit from a set of samples X. X is also called design matrix, where each row is a concrete sample x and each column represents a feature v of all considered features V. Each class i, of all classes I that shall be represented in the system, needs to be present at least once in the training set. During training, the classifier f infers the underlying statistical distribution of the features with respect to the class labels I. That is, the machine learning system tries to infer the mapping $i = f(x)$. The classifier cannot learn a class distribution that is not provided during training, and hence supervised classification inherently operates on a closed-set of data, i.e., no specific labels can be assigned to an unknown class. All samples used for training are presented in a tuple (x, i). Hence, in the scenario of an unknown camera model j, we propose to predict its make i: since there are far fewer makes than models, and it can be relatively safely assumed that cameras of the same make are present in the dataset.

Thus, the class labels I for classification are the camera makes, e.g., Nikon, Apple, or Samsung. To formalize this conclusively, each model j is a subset of all models J of one particular make i. Fig. 2 visualizes this. The blue and green boxes represent enclosed sets of models that belong to exactly one make. The set of models are disjoint.

In a real-world scenario, it can be assumed that the camera makes and models are quite unevenly distributed due to different market shares. Such an imbalance is also reflected in our collected dataset. Sample imbalance is a known challenge to several classifiers. In preliminary experiments, extremely randomized forests (Geurts et al., 2006) performed quite well, which are used throughout the experiments in this paper. The forest consists of 100 trees. Nodes in the trees are split during training based on the gini impurity measure (Hastie et al. (2009)). This criterion measures how clear the classes are split

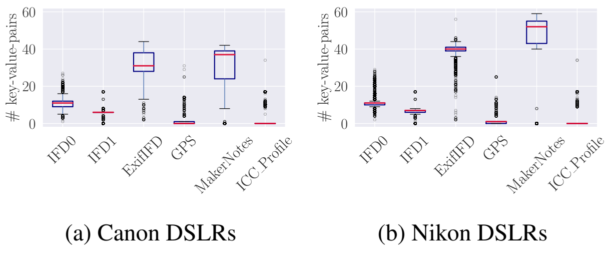


Figure 3: Aggregated number of key-pair-values per directory for two makes. Especially the directories `ExifIFD` and `MakerNotes` exhibit considerable differences between the two makes.

on a given decision boundary. The gini impurity equals zero if all samples belong to only one class in a node. Such a node becomes a leaf. We only used fully grown trees, e.g., all leafs are pure. Fully grown trees may be suboptimal in other classification tasks, since they are prone to overfitting. However, the header data that we are operating on is in two ways different from other classification tasks. First, it is almost categorical in its nature, particularly for makes that barely vary the header across models. Second, the imbalance of the sample set may lead to a suppression of underrepresented models. Both factors together benefit from a classifier that ensures the actual representation of each individual sample instead of interpolating a smoother decision boundary that may potentially "swallow" minority models. We used python's scikit library to work with decision trees.

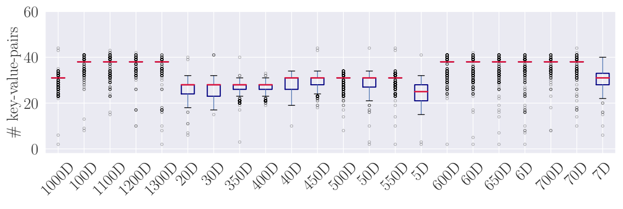
4. Variability of Header Information

Our hypothesis is that the header information between models from the same make are similar but vary significantly enough across models of different make to distinguish them. The range of variability of the header information is illustrated in this section. To this end, we study intra-make and inter-make similarities and differences in the header features. This illustration supports the understanding and interpretation of the classifier-based prediction in the subsequent sections.

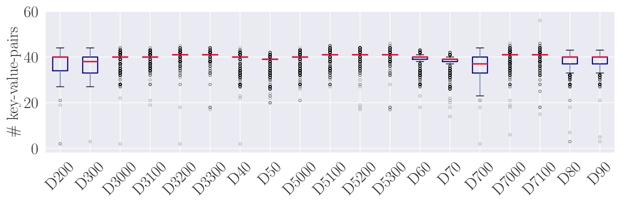
Kee et al. (2011) showed on DSLR cameras and compact cameras that many headers are unique to a particular model. However, neither the specific differences between non-matching headers have been studied, nor whether these differences are within makes or across makes. However, the variability of header information directly determines the difficulty for predicting the association of a model to a make. This variability is illustrated for EXIF-profile and color-profile metadata in Sec. 4.1, and for the encoding parameters in Sec. 4.2.

4.1. Variability of Metadata across Models

This first experiment characterizes the distribution of metadata from EXIF tags and the color profile. To this end, we select from our dataset 22 DSLR models from cameras by Canon and 19 DSLR models from cameras by Nikon. For each model, 1000 images are randomly drawn. During that sampling, it was ensured that the EXIF field `Exif:Software` is empty, i.e., that there are no obvious traces of additional processing.



(a) Canon DSLRs

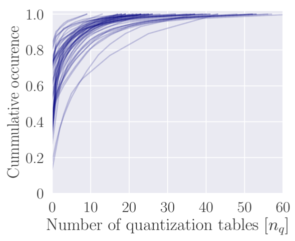


(b) Nikon DSLRs

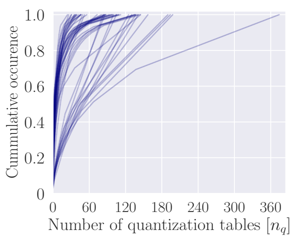
Figure 4: Box-whisker-plots for number of key-value-pairs in `ExifIFD` directories per model.

For these images, the metadata features are calculated, i.e., the number of entries per metadata directory are counted. The distribution of these values are shown in Fig. 3 on the left (Canon) and on the right (Nikon). These box-whisker-plots are bounded by the lower quartile (Q_1) and upper quartile (Q_3), the so-called inter-quartile distance. A smaller box indicates a denser distribution. The red bar within the box denotes the median of the distribution. The whiskers above and below the boxes extend up to $1.5 \cdot (Q_3 - Q_1)$, or the last datapoint on this side, e.g. if minimum or maximum are within this range. For both makes, the distributions are quite compact for the directories `IFD0`, `IFD1`, `GPS`, and `ICC_Profile`. The directory `ExifIFD` is also very compact for Nikon cameras, but not for Canon DSLR cameras. For both makes, also the median number of entries in `ExifIFD` differs. The distribution for Canon spreads between 28 and 39 key-value-pairs, while it is compactly centered around 40 key-value-pairs for Nikon, with only few outliers. Both figures also show significant differences in `MakerNotes`. This exhibits major variations both within the same make, but also across makes, where Nikon cameras show across models considerably more key-value pairs than Canon. We consider these findings as a first indication that differentiation of makes is possible with such a metadata distribution, although there may be considerable variations within one make.

Figure 4 provides a closer look at the distribution within the directory `ExifIFD`. These box-whisker plots show the key-pair-values distributions split per model, with Canon models on top and Nikon models on bottom. This representation shows correlations between the models of one make. For Canon, one group of cameras consistently has 38 key-value-pairs in the `ExifIFD` directory. The remaining Canon models are also similar, but less compactly clustered, with 20 to 30 key-value pairs. Thus, Canon's `ExifIFD` directory appears to be well representable with two clusters, which explains the somewhat larger standard deviation for the summary statistics in the previous Fig. 3. Conversely, the Nikon models on the bottom of Fig. 4b overall

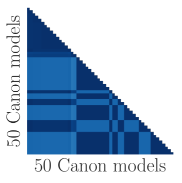


(a) 50 Canon Models

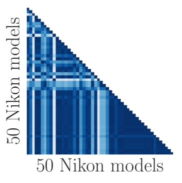


(b) 50 Nikon Models

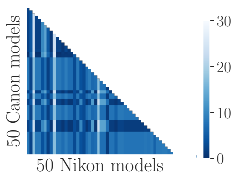
Figure 5: Occurrence of specific quantization tables within a given model. Each line represents a Canon model (left) or Nikon model (right). The x-axis indexes distinct quantization tables, sorted by their relative frequency per model. The y-axis shows the cumulative relative frequency of quantization tables per model. Nikon models exhibit a much larger variety of quantization tables within a model than Canon models.



(a) RMSE among Canon models.



(b) RMSE among Nikon models.



(c) RMSE between Canon and Nikon.

Figure 6: Root mean square error (RMSE) between quantization matrices of luminosity within 50 Canon models (left) and 50 Nikon models (middle). RMSE between 50 paired Canon and Nikon models (right).

exhibit a somewhat smaller variation in the median, with values between 39 and 41. Interestingly, the median is for many models at the boundary of the quartile boxes, which indicates a skewed distribution of the values.

We interpret this initial study as an indication that header information is able to distinguish between makes, but models may form clusters within one make. Additionally, each model exhibits a considerable number of values that are far away from the median value, which indicates that a lookup table, as used in previous work, e.g., Kee et al. (2011), is probably not able to fully cover the metadata variations that may be observed on real data. Hence, we argue that a classification-based approach is likely better suited to reflect such within-class variations.

4.2. *Variability of Encoding Parameters across Models*

We now aim at characterizing the variability of JPEG quantization matrices, using again models from Canon and Nikon. The number of models is expanded to 50, with 100 to 1000 randomly sampled images per model. These models include both DSLRs and compact cameras. As in the previous experiment, only clean images with an empty EXIF field `Exif:Software` are used in order to exclude reprocessed images. On all images of one model, the quantization matrix for luminosity is extracted. It turns out that images from all models have variations in their quantization matrices, which can be explained, e.g., with varying camera settings or potentially cameras that optimize the quantization tables to suit the image content best. The cumulative distribution of these matrices is shown in Fig. 5 for Canon (left) and Nikon (right). Both plots consist of 50 curves, where one curve indicates the cumulative distribution of quantization matrices for one model. The x-axis denotes the number n_q of different quantization matrices, the y-axis the cumulative percentage of images with the subset of n_q matrices. Thus, if a curve is close to the y-axis, then most of the quantization matrices of this model are identical. A curve that notably extends to the right exhibits many quantization matrices, and this variation affects a large percentage of the images.

When comparing the plots for Canon and Nikon models, it turns out that several Nikon models use a considerably larger set of different quantization tables. One Nikon model even showed 375 different quantization tables. The median number of quantization tables for Nikon models is 91. For the Nikon model with the most identical tables, only 38% of images shared the most common quantization table. Conversely, the most diverse Canon models had 64 different quantization tables, and several Canon models exhibited identical tables for more than 80% of the images.

Some of our data points may still have been processed by additional software, since our data collection protocol does not perfectly exclude such cases. Nevertheless, even if some of these matrices stem from external software, the overall statistics indicate that quantization matrices can exhibit a surprisingly broad distribution. In light of these findings, results in earlier work on header-based source identification (Kee et al., 2011) might somewhat too optimistic, as this work excluded rarely occurring quantization matrices. Conversely, we argue that this diversity of quantization matrices should be explicitly considered to minimize false negative assignments.

We further illustrate the quantitative differences between the quantization matrices within and across different makes. To this end, we select from each model the most frequently occurring luminance quantization matrix. In one experiment, we compare these quantization matrices within the 50 Canon models, in a second experiment within the 50 Nikon models, and in a third experiment between the 50 Canon and 50 Nikon models. We decided for the root mean squared error as metric to quantify the distances between pairs of vectorized matrices. The root mean squared error is equivalent to the euclidean distance between two vectors, and therefore, is a convenient metric to determine their distances (Han et al. (2011)) .

These distances are visualized in Fig. 6 in color-coded adjacency tables. The x- and y-axis consists of 50 rows and columns. The distances between model i_1 and i_2 is color-coded at position (i_1, i_2). Stronger saturation indicates more similar matrices.

Figure 6a shows the distances within models of Canon. Predominantly, these matrices are all relatively similar, indicated by the overall high saturation. The identical distances for many combinations indicate that there exist few clusters of identical matrices. Figure 6b shows the distances within models of Nikon. This figure is much more heterogeneous, due to the larger variation of Nikon quantization matrices. Also the absolute distances are in some cases significantly larger, with root mean squared errors of up to 28. Figure 6c shows the distances of models of Canon compared to models of Nikon. Here, the overall distances are larger, and also quite varied due to the diversity of the Nikon quantization matrices. We interpret these findings as another indication for the possibility to associate makes by quantization matrices.

		# Images per model			
Make	Models	Min.	Med.	Max.	Total
Canon	50	101	422	1000	25643
Nikon	50	102	264	1000	21038
Apple	25	188	1000	1000	19436
Sony	50	109	269	1000	17033
Olympus	50	100	205	936	13679
Panasonic	50	100	195	1000	12977
Samsung	30	101	136	330	5007
HTC	8	104	272	691	2512
Ricoh	5	117	158	331	995

Table 2: Overview of samples used for training the classifiers. The column "Models" shows the number of models per make. The minimum (Min.), median (Med.), and maximum (Max.) number of images across models are listed, together with the total number of samples. The imbalance in the number of images per make aims to mimick real-life distributions.

4.3. *Classification of Makes of Unseen Models*

In this experiment, we aim to exploit the reported observations in an automated classification of the camera make from an unseen camera model.

4.3.1. *Data Set, Classifier Training, and Performance Metrics*

We expand the evaluation data set from the previous subsection to a total of 9 makes. To reflect the real-life circumstance that manufacturers do have different numbers of models on the market, we did not limit the number of models per brand to the minimum number of models the weakest manufacturer (Ricoh, with 5 models) has. Instead, we randomly sampled up to 50 models per make, or the most number of samples available for that make. This large variation in models per make results in a challenging class imbalance by a factor of 10. Each model consists of at least 100 images, and if more than 1000 images are available, then up to 1000 images are randomly drawn from the pool. This varying amount of images again increases the class imbalance. Analogously to the previous experiments, care was taken that these images do not contain indications of additional software processing in the `Exif:Software` tag. In addition to this constraint, we linked back the images to the original Flickr users, and ensured that only one image per model and Flickr user is in the data set to avoid potential data bias. Table 2 provides an overview of the data distribution for this experiment, including the minimum, median, and maximum number of images per model, and the total number of images per make.

The data is split in a training set and a test set. The test set consists of all the samples of one particular model per make. The training set contains all samples of the remaining models. In Fig. 2 this is schematically illustrated by the picked colors. All samples of models shaded blue are taken for training, samples of the one model shaded green, is taken for testing. Training and test set form disjoint sets. Thus, the evaluated classifier has never seen a sample of a model under test during training.

The extremely randomized forests classifier is trained on four combinations of the feature vectors. These combinations consist of a) the metadata directory histograms, b) the luminosity quantization matrices, c) a concatenation of luminosity and chromaticity quantization matrices, and d) a concatenation of the metadata directory histograms and the luminosity quantization matrix. The classifier is fitted to the training data with the task to predict the make. During evaluation, the classifier is tested with the samples of the unseen model.

We report the accuracy as a performance metric of the classifier. Accuracy denotes the percentage of images for which the model is correctly identified belonging to its make. Accuracy can be calculated with respect to all makes or individually per make. However, in the case of class imbalance, the accuracy with respect to all makes is dominated by the most frequent classes. Hence, we report the accuracies per make to provide a more complete picture of the actual performance.

4.3.2. *Results and Discussion*

Figure 7 compares the performance of the extremely randomized forest classifier on the four feature sets. From left to right, these are the metadata directory histograms, the luminosity quantization matrices, concatenated luminosity and chromaticity quantization matrices, and concatenated metadata directory histograms and luminosity quantization matrix.

All figures show box-whisker-plots for the nine evaluated makes. The plot shows for each make the distribution of accuracies per model of that make. Overall, the accuracies for the first and last feature vectors are remarkably high, with almost all median accuracies beyond 90%. These experiments use the metadata directories as features. The performances of the plots in the middle are considerably lower. These experiments use the quantization matrices as features.

In the results that only use the metadata directories in Fig. 7a, Apple and HTC models are very reliably associated with their make. Ricoh models perform worst, but their median accuracy is still at about 90%. Ricoh is also the class with the fewest models and samples. Overall, all makes can be well associated with this feature set. In Fig. 7b, the luminosity quantization matrices perform surprisingly poorly. All median accuracies are considerably lower compared to the metadata directory features. Samsung devices exhibit the worst performance with a median accuracy that is only slightly above 20%. Apple still performs best, but the accuracy is also slightly lower than when using the metadata directory features. Also the variance of the accuracy increases, indicated by the larger boxes. In Fig. 7c, the addition of chromaticity quantization matrices to the luminosity matrices barely improves the results. A slight improvement can be seen for Sony models, but overall the results are well comparable to features that only use luminosity quantization matrices. Finally, Fig. 7d shows that the combination of features from metadata directories with luminosity matrices performs overall comparably to features from metadata directories alone. Median accuracies are typically in the range of 90% and beyond. Ricoh and Samsung models perform somewhat worse compared to using just the metadata directories.

In summary, features that only use the metadata directories perform best. This is in line with the preliminary observations in Sec. 4 on the variability of metadata information and quantization matrices: the metadata structure is relatively robust across various models of manufacturers, such that a classifier can be successfully trained for source association with an accuracy of 90% and higher. We find these results quite encouraging, particularly when considering that the images are retrieved from an uncontrolled public source, and that some images may be polluted by influencing factors outside of our control.

It is surprising to observe that the quantization tables perform considerably worse. One reason for this observation is certainly that we did not restrict the image set to the most common quantization matrices, as it has been done in previous work. Another reason might be that this 64- or 128-dimensional feature vector is much more complex than the metadata directories, and it is more difficult to identify the discriminating features. Moreover, our feature extraction and classification pipeline does not exploit the special structure of the quantization matrices: The entries of a quantization matrix typically reside on a lower-dimensional manifold, since low-frequency quantization factors are by tendency smaller than high-frequency quantization factors. This particular property could be used to efficiently summarize the quantization matrices in a considerably lower-dimensional feature space in which each individual dimension is more informative. This could in particular help to summarize the multitude of quantization matrices within one make, as observed in Sec. 4.2 for Nikon models.

We consider the comparably weak performance of the quantization matrices also to be the prime reason why the combined feature vector in Fig. 7d does not exceed the performance of metadata directory features. This feature combination might also benefit from a mapping of the quantization features to a lower-dimensional space. We will investigate such advanced feature representations in our future work.

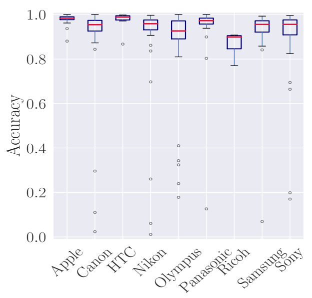
4.4. *Classification of Makes on Postprocessed Images*

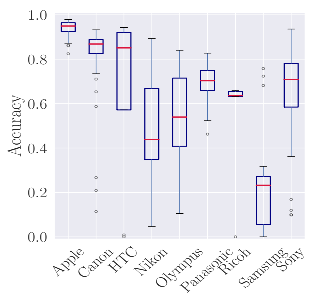
In practice, images are oftentimes post-processed, for example through an app on a smartphone or with a desktop PC software. Such a processing is likely to modify the header of an image, which increases the difficulty to associate an image to its model or make. This practically important question has to our knowledge not yet been addressed in the literature.

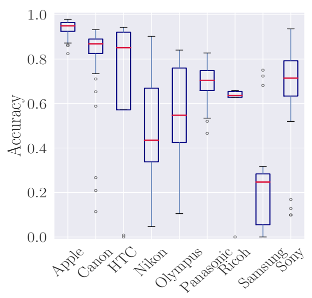
4.4.1. *Data, Classifier Training and Performance Metrics*

This experiment takes a slightly different perspective on the open-set problem: We train an extremely randomized forest on the full data set from the previous experiment. These images are carefully cleaned to exclude images that have undergone processing in external software packages. Then, we test on images where the EXIF field `EXIF:Software` states the name of a processing software, and aim to still predict the camera make that is stated in the `EXIF:Make` field.

More in detail, the testing data consists of images that have an entry in the field `EXIF:Software` that can be linked to a


(a) Features are number of key-value-pairs per directory.


(b) Features are entries of greyscale quantization matrix.


(c) Features are entries of both quantization matrices.

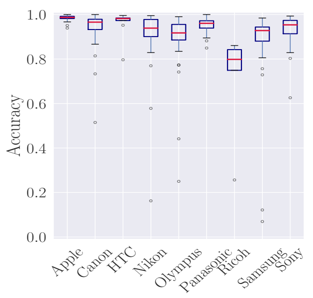
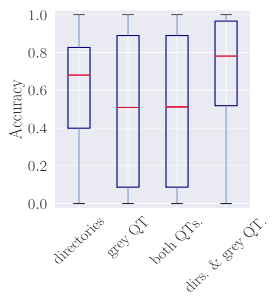

(d) Directories and greyscale combined.

Figure 7: Distribution of accuracies per make on four combinations of the feature sets.


(a) Accuracy on images that are postprocessed by apps.

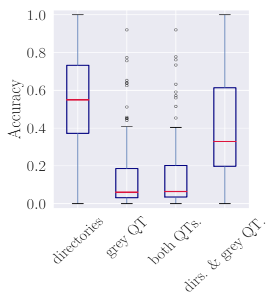

(b) Accuracy on images that are postprocessed by Desktop software.

Figure 8: Performance for predicting the make using four combinations of the feature sets on postprocessed images.

known software package, and hence we identify it as modified after acquisition. The names of the software tags are normalized as described in Sec. 3.1. We manually categorize the software entries in this dataset by their software platform, namely app-based software and desktop-based software. Tab. 3 shows examples of software packages in both categories. This categorization is in some details unsharp, as there are some pairs of desktop software with a companion app, or cloud services that serve both platforms. Nevertheless, upon manual inspection, we realized that most apps offer only simple filters to postprocess an image, while desktop-based software packages typically provide more sophisticated tools.

In total, the dataset consists of 121 desktop software packages and 27 apps. We conjecture that the desktop category is larger because the collected images date back until 2008, and because flickr is a platform for photographers, who may prefer to postprocess images on a desktop PC.

To quantitatively measure the performance, we report distributions of accuracies on the set of postprocessed images.

4.4.2. Results and Discussion

First, we report the accuracies across all images with an `Exif:Software` field, separated by desktop software and apps. These results are shown in Fig. 8. On the left, results for apps are shown, and on the right for desktop software. The results are calculated for all four combinations of the proposed features. Overall, the median accuracies range between 50% and about 75% for apps, and between about 10% to slightly beyond 50% for desktop software. Nevertheless, the guessing chance on nine makes is 11.11%. This baseline is easily exceeded across apps and desktop software for features from the metadata directories, either as stand-alone features or in combination with luminosity quantization matrices. In all experiments with a reasonable accuracy, the inter-quartile distances are quite large, which indicates that the confidence that can be put on individual results is relatively low. Overall, it is quite reasonable that the performance in this experiment is lower than in the previous experiment. The classifier is evaluated on images that underwent a potentially large modification in the processing software.

Postprocessing of apps simplifies the association of an image to the camera make. We hypothesize that apps have a stronger tendency to reuse library functions from the smartphone operating system stack, which effectively has a smaller impact on the JPEG headers compared to processing in a desktop software. Moreover, even desktop sofware leaves characteristic traces in the metadata that allows to associate images to the original make with a median accuracy of about 50%.

Table 3 lists the apps and desktop packages that perform best and worst for both categories. For the apps, it is interesting to note that the apps from the same organization may have quite different accuracies. Specifically, the Flickr app is extremely well recognizable in many versions with an accuracy of 100%, but “Flickr for iPhone” completely fails with an accuracy of 0%. Similarly, several versions of Camera+ can be recognized with accuracies between 77% and 90%, but version “2863” is only recognized with an accuracy of 43%.

Desktop software is more likely to use a software stack that is considerably different from the camera software stack. Additionally, we observed that several desktop software packages rewrite major parts of the header. An example for this behavior is studied on images from “Adobe Photoshop Lightroom 5”. Figure 9 shows the distribution of key-value pairs per directory on images with this entry in the `EXIF:Software` field. The images in this experiment come from any make or models available, so the original distribution of header information can be assumed to be quite broad. Nevertheless, af-

App

Rank	Software	Version(s)	Acc.
1	Flickr	1, 2, 6060, 3622	1.00
5	Mobile Fotos	_, 2, 1006	[0.97–1.00]
8	Instagram	_	0.95
9	Camera+	3, 6, 7, 9	[0.80–0.90]
13	PicsArt	_	0.80
14	VSCO	_	0.78
15	Camera+	5	0.77
	...		
23	Camera+	2863	0.43
24	Pixlr	_	0.40
25	Camera360	_	0.40
26	Snapseed	_	0.18
27	Flickr	for iPhone	0.00

Desktop

Rank	Software	Version	Acc.
1	Capture NX-D	1	1.00
2	Microsoft Windows Photo Viewer	6	0.93
3	Nikon Transfer	2	0.93
4	ViewNX-i	1	0.91
5	Phatch	_	0.90
6	Photoshop Express	3	0.88
7	DXO Optics Pro	v5	0.87
	...		
116	ACDSee Ultimate	10	0.08
117	Elements Organizer	14	0.07
118	ACDSee Pro	8	0.06
119	Capture One	8	0.06
121	Imagen digital ACD Systems	_	0.00

Table 3: Examples of software split into apps (top) and desktop (bottom). Listed are the software packages that performed best and worst in header association, together with their accuracy. Comparable results of multiple versions of the same software are summarized in one row. The software versions are listed in the second column. The underscore denotes versions without identifier.

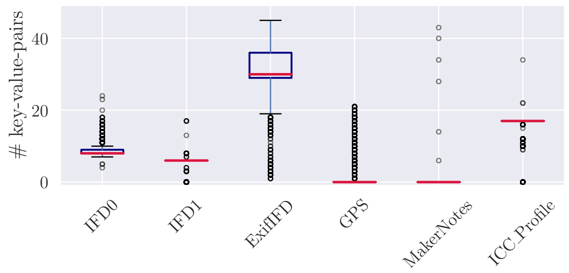


Figure 9: Distribution of key-value pairs, after the images were edited with "Adobe Photoshop Lightroom 5".

ter processing in Adobe Lightroom, all directories IFD0, IFD1, GPS, MakerNotes, and ICC_Profile are virtually identical in size. This indicates that Adobe Lightroom rewrites these entries. Interestingly, the directory ExifIFD contains a larger inter-quartile distance and an overall larger spread. We hypothesize that this directory may still contain some of the original header information from different models and makes. The combination of erased and preserved header information can potentially be useful for further software fingerprinting. We will investigate this direction in our future work.

5. Conclusions

In this paper we investigated source identification from file headers of JPEG images. Headers are extremely fast to read out and process, and are as such an ideal tool for fast screening of large amounts of multimedia data. However, a pathological issue in source identification is to maintain an up-to-date model database, given that new camera models appear every month, and updatable software stacks modify the camera characteristics. Hence, we propose to consider source identification as an inherent open-set problem. To mitigate this issue, we propose to use known models to predict the make of a camera.

Towards this goal, we collected a large database of images from the website Flickr, and we proposed two sets of features: a histogram over the number of metadata directory entries, and the quantization matrices of JPEG files. Three major experiments were performed to study the possibility to predict the camera make from header information. First, we illustrate the variability of header information across makes. Second, we classify camera makes with combinations of the proposed features. It turns out, that the histograms over the metadata directories are an excellent tool to associate an image to the camera make. Third, we study the impact of apps and desktop postprocessing software on the same task. In this considerably more difficult task, images with postprocessing by apps can in many cases still be well associated, with a median accuracy of about 75% for a combination of histogram and quantization features. However, we also show that desktop software has a strong tendency to rewrite major portions of the header, with a median accuracy of only 55%. Nevertheless, this result is still well above guessing chance of 11.11%, which suggests that more sophisticated features may yield further improvements. This potentially includes hand-crafted features that exploit specific data properties, or automatically learned features from deep neural networks. The fact that manufacturers like Nikon deploy different, possible image content sensitive, quantization matrices may open another interesting research direction: For example, it may be possible to find a low-dimensional manifold where these matrices reside to simplify header association from encoding parameters.

Acknowledgements

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

Bayar, B., Stamm, M., 2018. Towards Open Set Camera Model Identification using a Deep Learning Framework, in: International Conference on Acoustics, Speech and Signal Processing, pp. 2007–2011.

Bayar, B., Stamm, M.C., 2017a. Augmented Convolutional Feature Maps for robust CNN-based Camera Model Identification, in: International Conference on Image Processing, pp. 4098–4102.

Bayar, B., Stamm, M.C., 2017b. Design Principles of Convolutional Neural Networks for Multimedia Forensics. Electronic Imaging , 77–86.

Bondi, L., Baroffio, L., Guera, D., Bestagini, P., Delp, E.J., Tubaro, S., 2017. First Steps Toward Camera Model Identification With Convolutional Neural Networks. Signal Processessing Letters 24, 259–263.

Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. Machine Learning 63, 3–42.

Gloe, T., 2012. Forensic Analysis of Ordered Data Structures on the Example of JPEG Files, in: International Workshop on Information Forensics and Security, pp. 139–144.

Gloe, T., Fischer, A., Kirchner, M., 2014. Forensic Analysis of Video File Formats. Digital Investigation 11, 68–76.

Gloe, T., Kirchner, M., Riess, C., 2013. How we learned to stop worrying about content and love the metadata. Technical Report. 1st IEEE IFS-TC Image Forensics Challenge, category "Format-Based Attacks".

Goljan, M., Fridrich, J.J., Filler, T., 2009. Large Scale Test of Sensor Fingerprint Camera Identification, in: Media Forensics and Security I, IS&T-SPIE Electronic Imaging Symposium, San Jose, CA, USA, January 19-21, 2009, Proceedings, pp. 0I–01–0I–12.

Güera, D., Baireddy, S., Bestagini, P., Tubaro, S., Delp, E.J., 2019. We need no pixels: Video manipulation detection using stream descriptors. CoRR abs/1906.08743. URL: `http://arxiv.org/abs/1906.08743`.

Hamilton, E., 1992. JPEG File Interchange Format. `https://www.w3.org/Graphics/JPEG/jfif3.pdf`.

Han, J., Kamber, M., Pei, J., 2011. Data Mining: Concepts and Techniques, 3rd edition. Morgan Kaufmann. URL: `http://hanj.cs.illinois.edu/bk3/`.

Hastie, T., Tibshirani, R., Friedman, J.H., 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition. Springer Series in Statistics, Springer. URL: `https://doi.org/10.1007/978-0-387-84858-7`, doi:`10.1007/978-0-387-84858-7`.

Iuliani, M., Shullani, D., Fontani, M., Meucci, S., Piva, A., 2019. A Video Forensic Framework for the Unsupervised Analysis of MP4-Like File Container. Transactions on Information Forensics and Security 14, 635–645.

Júnior, P.R.M., Bondi, L., Bestagini, P., Tubaro, S., Rocha, A., 2019. An In-Depth Study on Open-Set Camera Model Identification. arXiv preprint arXiv .

Kee, E., Johnson, M.K., Farid, H., 2011. Digital Image Authentication from JPEG Headers. Transactions on Information Forensics and Security 6, 1066–1075.

Lukás, J., Fridrich, J., Goljan, M., 2006. Digital Camera Identification from Sensor Pattern Noise. Transactions on Information Forensics and Security 1, 205–214.

Marra, F., Poggi, G., Sansone, C., Verdoliva, L., 2017. Blind PRNU-Based Image Clustering for Source Identification. Transactions on Information Forensics and Security 12, 2197–2211.

McKay, C., Swaminathan, A., Gou, H., Wu, M., 2008. Image Acquisition Forensics: Forensic Analysis to Identify Imaging Source, in: International Conference on Acoustics, Speech and Signal Processing, pp. 1657–1660.

Mullan, P., Riess, C., Freiling, F., 2019. Forensic Source Identification using JPEG Image Headers: The Case of Smartphones. Digital Investigation 28, S68–S76.

Pennebaker, W.B., Mitchell, J.L., 1992. JPEG: Still image data compression standard. Springer Science & Business Media.

Stamm, M., Bestagini, P., Marcenaro, L., Campisi, P., 2018. Forensic Camera Model Identification: Highlights from the IEEE Signal Processing Cup 2018 Student Competition [SP Competitions]. Signal Processing Magazine 35, 168–174.

Wallace, G.K., 1992. The JPEG still picture compression standard. Transactions on Consumer Electronics 38, xviii–xxxiv.

Wronski, B., Garcia-Dorado, I., Ernst, M., Kelly, D., Krainin, M., Liang, C.K., Levoy, M., Milanfar, P., 2019. Handheld Multi-Frame Super-Resolution, in: Transactions on Graphics, pp. 28:1–28:18.