Digital Investigation 28 (2019) S68–S76

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Forensic source identification using JPEG image headers: The case of smartphones

Patrick Mullan*, Christian Riess, Felix Freiling

IT Security Infrastructures Lab, Friedrich-Alexander University Erlangen-Nürnberg, Germany

ARTICLE INFO

Article history:

Keywords:
Image metadata forensics
Image forensics
EXIF tags
Quantization matrix
Source identification

ABSTRACT

A common problem in forensic investigations is the identification of the source of multimedia data, i.e., determining the model, make or individual device that recorded media content. In contrast to methods based on sensor noise, source linkage based on header information of media items allows for easy automation. Such header information involves metadata like EXIF tags and the parameterization of the JPEG algorithm. While traditional digital cameras typically had a fixed software stack that makes it straightforward to fingerprint a device, modern mobile devices vary considerably in their software stack over time. We perform a large-scale study of JPEG header information from Apple smartphones to investigate the effect of this development on the possibility to perform source identification. Our analysis shows that identification of the concrete hardware is much harder for smartphones than it is for traditional cameras. However, identification of software stack, particularly the operating system version and selected apps, is well feasible.

Introduction

One important goal of digital multimedia forensics Farid (2016); Redi et al. (2011) is to reconstruct the provenance of an image or video, i.e., to narrow down the circumstances under which the image or video was originally recorded or subsequently processed. In this field, a lot of research activity has been devoted to the analysis of the actual (pixel) content of the multimedia object. A seminal paper by Lukáš et al. (2006) showed that slight manufacturing inaccuracies of the sensor cells allow to calculate a device-specific sensor fingerprint by extracting the fixed pattern noise of an image. This fingerprint can subsequently be used to identify the camera that was used to take a particular picture. A substantial body of follow-up work has successfully confirmed and refined these findings Goljan and Fridrich (2008); Goljan et al. (2010); Li (2010); Stamm et al. (2013). However, source identification based on sensor noise requires large volumes of image data and particular expertise in signal processing to choose appropriate statistical methods and interpret the results. These methods therefore require time and are restricted to experts operating in forensic laboratories.

Next to having high confidence in source identification, in practice it is often equally important to *quickly and automatically* establish medium confidence leads to guide an investigation. For example, when acquiring pictures from Internet sites such as Flickr, investigators are often not interested in the particular camera that took an image but to quickly curb the set of possible devices. This is especially relevant for pictures taken by personal devices such as smartphones, since the exclusion of particular devices immediately narrows down the set of suspected individuals. This can be done by using metadata from the *image container*. It is well known, that such metadata can be relatively easily modified, for example social media sites regularly change the specifics of the syntactic representation of a media item Ng (2013); Giudice et al. (2017). Therefore, image metadata is often overlooked since it is considered to have (almost) no probative value.

In this paper we argue that this is not necessarily the case and show that, especially and particularly for images taken by smartphone cameras, image metadata, like EXIF tags, and parametrization of the employed JPEG encoding algorithm can be used to quickly and automatically establish reliable source identification.

Related work

Compared to the large body of work on *content-based* methods, evidence taken from the *media metadata* received surprisingly little

* Corresponding author.
E-mail address: patrick.mullan@fau.de (P. Mullan).

https://doi.org/10.1016/j.diin.2019.01.016

attention in the past. In general, two types of such metadata exist: (1) information on the *image encoding* like quantization tables in JPEG, and (2) general information in the *media container*. Those general information include the widely known EXIF data. On top, also other segments in the file container can exist, that are not necessarily standardized by EXIF, like copyright information or colorprofiles.

Several works investigated JPEG metadata for source identification: Ng (2013) studied the change in quantization tables of JPEG images by facebook. Giudice et al. (2017) and Castiglione et al. (2011) studied a similar problem, but in a slightly wider scope, including resizing, compression, renaming and metadata alterations on several social media sites. Caldelli et al. (2017) investigated artifacts from double JPEG compression through Facebook, Twitter and Flickr to identify devices.

Focusing on traditional digital cameras from manufacturers like Canon, Sony or Nikon, the large-scale study by Kee et al. (2011) proposed to use encoding parameters as a set of features to determine the acquisition device of an image. More specifically, image dimensions, quantization and Huffman tables were combined into a highly indicative and stable digital camera fingerprint. Similarly, the sequence of JPEG data structures in the JPEG header was proposed by Gloe (2012) as a novel and reliable cue on the JPEG encoder.

Further, Gloe et al. (2013) investigated the header information of PNG images, and showed that converting an image from JPEG to PNG does not necessary erase all metadata. Which metadata is left depends on the tool converting the image.

In summary, source identification based on image metadata has been shown to work using automated classifiers. However, since digital cameras typically come with a fixed firmware that is not subject to any updates, it is unclear whether these findings extend to images taken by smartphones with their regular software and operating system updates.

Contributions

In this paper, we investigate the factors that influence metadata- and JPEG-encoder- based source identification of images taken by smartphone cameras. Due to its wide prevalence, we focus on images taken by Apple's iPhone devices.

Since its introduction, the iPhone has undergone several major developments in terms of hardware (e.g., iPhone 4 to today's iPhone X) and its operating system iOS. Furthermore, the applications with which users can take pictures on their iPhone are numerous. All these factors potentially influence image metadata and routines how an image is saved to disk. Compared to the ecosystem of traditional cameras, the variety of circumstances for image metadata grows considerably. As a result, source device identification is also considerably more complicated as in the case of traditional digital cameras. This is illustrated in Fig. 1: The classical distinction of source identification granularity (type, make, model, device) Kirchner and Gloe (2015) which holds for traditional cameras is disrupted by the varieties of the software stack which typically weakly correlates with the camera model. The software stack therefore represents an additional dimension for camera identification not known before. We show that JPEG header information, can be the used for successfully positioning an image in this dimension, i.e., associating an image to a particular version of the software stack.

Both factors together make it very challenging to use the media container to determine the origin of data that was distributed over social media. Nevertheless, for the broader scope of forensic activities, social media is oftentimes intentionally not involved. For example, consider a bank that hands out small loans based on

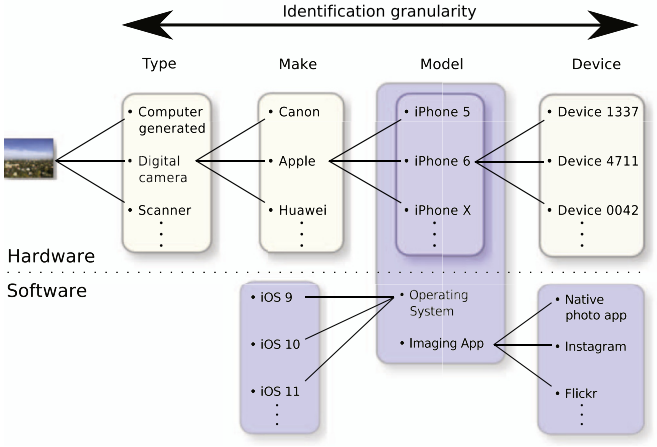

Fig. 1. The traditional (horizontal) granularity levels of forensic source identification must be extended by an additional dimension representing the software stack (in particular the operating system and the imaging app) for smartphones.

copies of salary statements of the past three months. In this case, it is reasonable to assume that a user directly takes pictures of these salary statements with a smartphone, and then electronically submits these pictures to the company. In this case, an automated metadata analysis, combined with smartphone fingerprinting, can be one component in a fraud protection software module. Another scenario may involve a metadata-based data clustering for police agencies after seizure of a smartphone or computer from a suspect.

In this work, we perform an in-depth analysis of JPEG container data. To limit the scope of the study, we focus on mobile devices from Apple, namely smartphones and tablets. Several earlier works focused on metadata-based fingerprinting of digital cameras Kee et al. (2011); Gloe (2012); Kirchner and Gloe (2015). We show that the rich and dynamic software environment on smartphones allows for a much greater variety of device-specific metadata configurations. Particularly impactful are the version of the operating system, but oftentimes also the specific apps that are used to acquire a picture.

More concretely, our contributions are twofold:

1. We collected a dataset for large scale evaluation on the robustness of metadata and compression parameters of smartphone images, with am attention on Apple devices. The dataset was downloaded from the picture sharing website Flickr, and after filtering and parsing, the dataset contained more than 200 000 images, of which 64 230 were from Apple devices.
2. We show that, due to the variety of the software stack, the classification of the concrete hardware is harder for the case of smartphones than for traditional cameras. On the above dataset, we perform experiments with an automated classifier to determine the provenance of an image. Using only the number of used EXIF headers and the value of the JPEG quantization tables as features, we show that several specific apps or operating system versions can be distinguished with high reliability.

Roadmap

This paper is organized as follows: The background on JPEG metadata is given in Sect. 2. The design of our study, the dataset and the applied methodology are presented in Sect. 3. Results from this data, and an evaluation of the automated classification are

presented in Sect. 4. We conclude this work in Sect. 5.

Background

To lay out the methodology of the present study, we first give some background on the components of a JPEG file.

Composition of a JPEG file

A JPEG-Image consists of multiple parts. The JPEG standard itself only defines the encoder and decoder of the pixel data Pennebaker and Mitchell (1992). The file format, which is in the focus of this study, was first defined in the "JPEG Interchange Format" (JIF). This standard specifies the container in which the encoded image is stored. However, JIF is rarely used, but instead the later developed "JPEG File Interchange Format" (JFIF) is used Hamilton (2004). JFIF specifies several more details and is therefore easier to implement than JIF. In parallel to JFIF, also the "Exchangeable Image File Format" (EXIF) was developed Electronics and Association (2016). Those two standards are nowadays the most widely used file formats for handling JPEG-encoded images.

Overall, the specification for JPEG encoding only ensures that all information is contained to successfully decompress the file. However, the exact choice of parameters, metadata, and the order of operations is left to the software implementation, and can therefore be used for fingerprinting a software stack.

JPEG encoding

JPEG compression transforms a raw RGB bitmap into a binary stream. First, the colors are transformed from RGB space to YCbCr space, with the ultimate goal of compressing the color channels Cb and Cr stronger than the perceptually more important luminosity channel Y. It is interesting to note that JIF allows free choice of the color space, while JFIF and EXIF explicitly require YCbCr Pennebaker and Mitchell (1992); Wallace (1992). After optional downsampling of the channels, non-overlapping blocks of 8×8 pixel are transformed into frequency domain with the Discrete Cosine Transform. The resulting 64 coefficients of a block are quantized using a 64-element table of quantization coefficients, which leads to a loss in image structure. Typically, the perceptually more important low-frequency components are quantized less strongly than the visually less important higher frequency components. The JPEG quality factor can be seen as a scaling factor to the quantization coefficients, where lower quality implies higher quantization factors. The quantized data is then further compressed using Huffman encoding to obtain the final bitstream.

The recommended baseline encoding scheme sequentially encodes the 8×8 blocks from left to right and top to bottom of the image Hamilton (2004). Notable variations are extended sequential, lossless, and progressive scanning Pennebaker and Mitchell (1992); Salomon (2007). Progressive scanning first encodes the lower frequency components of the whole image, then the higher frequency components. This allows a low-detail preview of the image without transmitting or decoding it completely.

Images files

The JFIF and EXIF standards for the image container build on JIF. A JIF file can be split into parts. One such part can for example be a quantization table, or a Huffman table. Each part starts with a specific marker. JIF also allows for so-called application markers (APPn), where JFIF uses APP0, and EXIF APP1. APPn markers can be populated with entries in form of key-value pairs. The values can be human-readable strings, or more complex structures like binary data. The APP0 segment defines pixel densities and pixel ratios, and an optional thumbnail of the actual image can be placed here. In APP1, the EXIF standard enables cameras to save further information about the image in so called EXIF-Tags. EXIF-Tags cover a huge range of optional information, many of them related to photographic settings of the camera when and where the image was taken. The information are subdivided in image file directories (IFDs).

Other metadata can also be stored in the file header. Extensible Metadata Platform (XMP) is also written in APP1 of the JIF File.[1] The Information Interchange Model by the International Press Telecommunications Council (IPTC-IIM-Standard, or just IPTC[2]) is saved in APP13. XMP and IPTC allow to provide additional information like keywords, copyright information or an editing history of the image. In APP2, an additional ICC color profile can be defined, to faithfully reproduce the colors in the image.[3]

Study design and dataset

We now describe the design of our study and the collected dataset. First, the considered JPEG header features are presented in Sect. 3.1. Then, we present the approach to data collection from Flickr in Sect. 3.2. Finally, we describe the used machine learning approach for source attribution in Sect. 3.3.

JPEG header features for this study

We perform this study on two families of features, namely the number of entries in the image file directories (IFDs) and the JPEG quantization tables. We also investigated the Huffman tables, but did not find them indicative for fingerprinting.

For this study, we consider the IFDs "ExifIFD", "IFD0", "IFD1", and "GPS". We also include the special directories 'ICC_Profile' and 'MakerNotes'. These are not part of the EXIF standard, but we observed that these are nevertheless oftentimes set. For each of these directories the number of key-value-pairs are counted resulting in a discrete number greater or equal to zero. The exact content of the key-value pairs in each of these IFDs is beyond the scope of this study, to avoid a tedious and potentially error-prone distinction of variant and invariant features. As will be shown later, the number of entries is already a quite robust feature for fingerprinting.

For the JPEG quantization tables, we only use the Y and Cb tables. This is based on the observation that in almost all JPEG implementations, the Cr and Cb tables are configured to be identical. To determine unique pairs of quantization tables, we concatenate these two tables and hash their $2 \cdot 64$ values. One pair of quantization tables can therefore be represented by one alphanumeric number, which we call QC-Table.

Kee et al. (2011) used in their earlier work also the Huffman tables for fingerprinting. Although an encoder could in principle compute a Huffman table for each image individually, it is our observation that most encoders prefer to use a fixed Huffman table, which could in that study be used for fingerprinting. However, in a preliminary experiment, we found that on our database, the Huffman tables were in the vast majority of cases identical, which is why we omitted this cue. For similar reasons, we also omitted the sequence of JPEG syntax elements that have been proposed in earlier work by Gloe (2012).

[1] XMP: https://www.adobe.com/devnet/xmp.html.
[2] IPTC-IIM-Standard: http://iptc.org/standards/iim/.
[3] ICC:https://docs.oracle.com/javase/6/docs/api/javax/imageio/metadata/doc-files/jpeg_metadata.html.

Data collection

The dataset for this study is acquired from Flickr. Via the API, we queried about one million images. One such query returns additional information on the respective image, which is evaluated before downloading the image itself. We used similar filter rules that were also applied in the study by Kee et al. (2011), which aim at performing a first keyword validation whether the image underwent any processing before uploading to Flickr. That is, we checked if metadata for the images was present at all, and whether the values for make and model are set in the EXIF data. If modification date and creation date are not closely together, or the creation date is missing, the image was also rejected. Additionally, we only considered images where we could successfully parse the date and time format. Examples of rejected date strings are representations that contained non-ASCII characters or apparently malformed strings like "2008:1127:27 10:53:". Additionally, we limited the data collection to images that were recorded between 2008 and 2018. It is worth noting that the date strings of a small number of images was found to lie in the future, e.g., with a year larger than 2018. Those images were also rejected, but these examples illustrate that not all date labels in the database may be perfectly accurate. Furthermore, we followed Kee et al. (2011) in blacklisting several known image processing software packages that can be found in the EXIF data under "EXIF:Software", which we considered an indication that the image underwent additional processing. We downloaded all images that passed these checks, such that the actual database consists of 432305 images. Additionally, we also recorded the owner of the upload so we can later filter images by users. That way, we identified 216852 unique users in our dataset.

Whenever we need to associate an image to a creation date, we use the EXIF field "DateTimeOriginal". This entry is the foundation for studying the evolution of JPEG header information over time. Using this entry, we study the distribution of source devices in the dataset. A plot, based on our dataset, for the relative distribution of images from eight camera hardware manufacturers over time is shown in Fig. 2. Smartphones recently became popular. Overall, it shows that smartphones are increasingly used as imaging devices in the Flickr community. Especially Apple got a much bigger share in images uploaded to Flickr in recent years. Other camera manufacturers like Kodak and Nikon seem to have less active users uploading images to Flickr. The numbers also indicate that most ($\sim$ 70%) of images on Flickr today seem to be recorded with cameras of only three manufacturers ('Apple', 'Canon', and 'Nikon'). Thus, we selected for this study the data from apple devices to study header information on a smartphone platform.

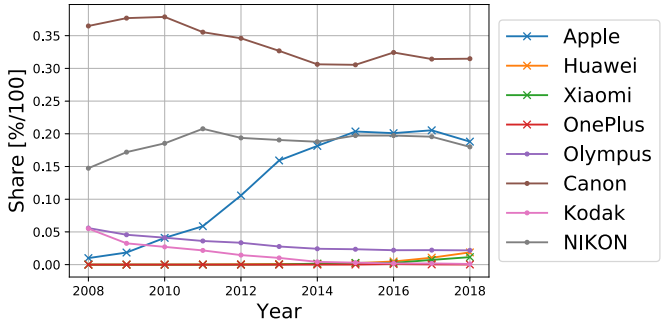

Fig. 2. Temporal distribution of images by manufacturer over a period of eleven years in our database. Graphs marked with a 'x' represent manufacturers of mobile phones and tablets, graphs with a '.' represent manufacturers of digital cameras. Overall, images from mobile devices are increasing. Identified Apple devices contributed a significant share of the images already between the years 2009–2016.

Header association via machine learning

We aim at linking an image to a particular model, make, or to a particular software from the header information. To encompass this, we select a subset of the database where "EXIF:Model", "EXIF:Make", and, if needed, "EXIF:Software" are set. We aim to use this information as ground truth for the classification. In general, we have no guarantee that this information, as downloaded with the images, is accurate. For example, it might be that an image has been edited with several software tools, but only one tool (not necessarily the last one) left information in the EXIF:Software field. To remove apparent wrong labels from the dataset, we performed several further filtering by image size. However, we are aware of the possibility that even the most careful data preparation will likely contain some noise in the label from internet-retrieved data as it is used in our database. In that sense, the reported results form are a conservative lower bound compared to a perfectly cleanly labeled set of data.

Note that model, make, and software can in general not be expected to map uniquely to a set of metadata and quantization matrices. For example, switching GPS tracking on and off can modify the metadata, or changing the quality settings between taking two pictures changes the quantization tables. Furthermore, it is also possible that the same set of metadata and quantization matrices can be found for various camera models, make, or for different softwares.

These ambiguities make it difficult to use lookup tables for source attribution as it has been done, e.g., in previous work by Kee et al. (2011). Instead, we use a machine learning approach to perform this task. The classification is done with a random forest. The features are the features stated in the previous section, i.e., the numbers of entries per directory and the quantization tables.

During training, the random forest generates an ensemble of decision trees. Each internal node of the tree learns a split on the feature space. The full tree thus performs a sequence of splits from the root node to the leaf nodes. Class membership is then determined by the relative frequency of classes in the leaf nodes. Randomness is introduced in two ways. First, not each tree obtains the full set of training samples but only a subset. Second, when finding a split, not all feature dimensions are presented in that node. This procedure is known to yield a robust classifier that is able to deal with outliers and noise, as it can be expected in our dataset Breiman (2001); Hastie et al. (2009). The Gini impurity criterion is used to select a feature for performing a split in a node Hastie et al. (2009). In all experiments with random forests, 100 trees were fully grown. We use 90% of the data for training, and 10% of the data in a separate test set for evaluation.

Evaluation

The evaluation consists of four parts. First, we perform a general analysis of the distribution of the header information of Apple smartphones over time in Sect. 4.1. Then, we investigate how well this header information can be linked to a smartphone hardware model in Sect. 4.2. This task is comparable to a previous study on digital cameras by Kee et al. (2011), but it can be seen that the smartphone JPEG headers provide only weak cues to the actual hardware. In Sect. 4.3, we repeat this experiment, but aim for classifying the version of the smartphone operating system instead. It turns out that the header information provide a much better cue on the smartphone software than the hardware. Finally, we show in Sect. 4.4 results for selected applications that left traces in the EXIF:Software tag, indicating that some apps leave their own individual traces that can be very reliably detected.

Temporal evolution of metadata information

We first examine how the EXIF metadata information changes over time. This is visualized in Fig. 3 for Canon EOS 450D cameras, i.e., a traditional digital camera, and in Fig. 4 for the smartphone iPhone 4S from Apple. Both figures are subdivided into five columns that represent the EXIF directories "ExifIFD", "IFDO", "IFD1", "GPS" and "MakerNotes". On the *x*-axis, the years from 2011 to 2018 are shown. On the *y*-axis, the number of entries per directory are shown. The heat map visualizes the relative frequency of the number of entries per directory, normalized for each of the years.

It can be seen in Fig. 3 that the distribution of entries in the Canon EOS 450D camera are indeed constant over time, which certainly benefits source identification. For example, consistently more than 70% of images contained 31 attributes in the directory "ExifIFD". The directories IFD1 and MakerNotes exhibit a bimodal distribution: either 6 or 0 attributes are observed in IFD1, and either 32 or 0 attributes are observed in MakerNotes. No GPS values were found, since the Canon EOS 450D does not contain a GPS tracker.

In contrast to that, the number of entries per directory greatly vary for the iPhone 4S in Fig. 4. For example, the number of entries in ExifIFD decreases in the year 2013 from 25 to 24 values. In 2014 it increases to 31 attributes, and in 2015 to a total of 32 vales. MarkerNotes shows a similar behavior. Added to the header in 2014, the number of entries rises until 2016 from 6 to 8 values. IFD0 and IFD1 show in all years distributions of the numbers of entries instead of a unique number. Overall, these numbers suggest that it might be considerably more difficult to associate the header information to the hardware platform of a smartphone.

The distribution of iPhone 4S EXIF entries over time can be explained by the operating system version. The EXIF data by images taken with Apple smartphones also reveals the iOS Version of the device when an image was taken. In Fig. 5, we represent the same data as previously in Fig. 4. However, the difference is that now the *x*-axis shows the iOS version instead of the time in years. Both figures are similar, but the distribution is much more aligned with the iOS version, which can be seen from the considerably more pronounced peaks in the distribution.

To further illustrate the relationship between hardware platform, operating system, and metadata information, we list the most frequent value of each EXIF directory per iOS version for a selection of hardware models in Table 1. It can be seen that the number of EXIF entries is mostly constant for one iOS version, i.e., between

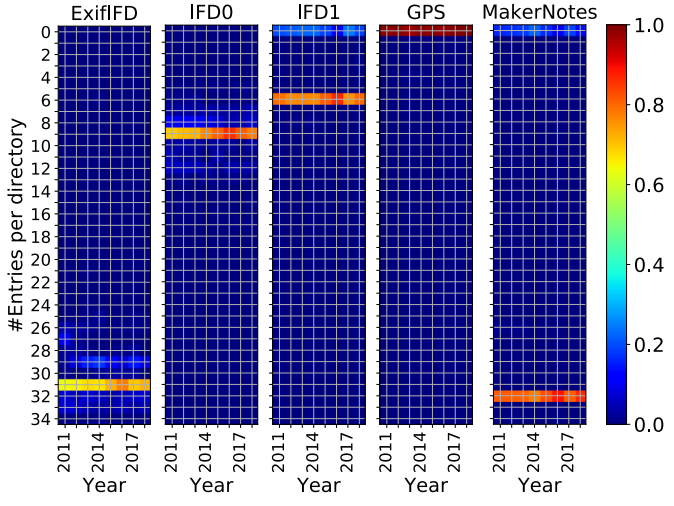

Fig. 3. Canon EOS 450D distribution of EXIF metadata over time. The metadata distribution is highly stable over eight years.

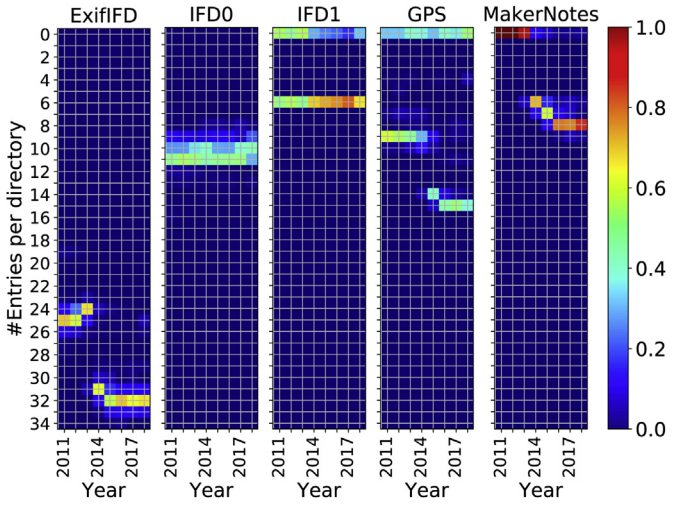

Fig. 4. Apple iPhone 4S distribution of EXIF metadata over time. The metadata distribution varies greatly with different software versions.

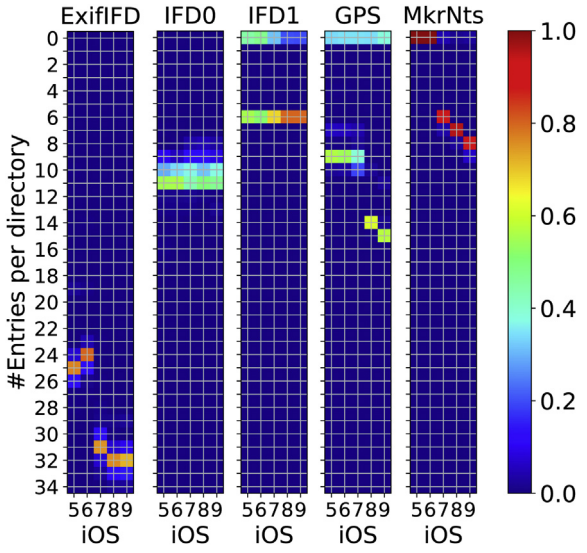

Fig. 5. Apple iPhone 4S distribution of entries per metadata directory over different iOS versions. The metadata distribution is very well aligned with the version changes of the operating system.

two horizontal lines. Exceptions, where two or more hardware platforms have differing numbers of EXIF entries for one iOS version are printed in bold face. Overall, this table indicates that the iOS version has significantly more impact on the distribution of EXIF values than the actual hardware platform. If at all, the directory of MakerNotes seems to be the most indicative for a Model given a specific iOS Version.

We also examined compression parameters, specifically quantization tables across iOS versions. We refer to the concatenated pair of quantization tables as pairs of QC-Tables. Fig. 6 shows the distribution of pairs of QC-Tables over iOS versions for an iPhone 4S (left) and an iPhone 5 (right). On the *y*-axis, the 10 most frequent pairs of QC-Tables are shown, and the columns are normalized to sum up to 1. The distribution shows that more than 80% of the images can be described with only two QC-Table-Pairs per iOS version.

To summarize these observations, Apple devices oftentimes updates their imaging routines, which leads to variations in the

Table 1
Most frequent number of EXIF entries per directory per iOS version and Model. In bold are cells, that have a varying number of values. Per user and iOS Version, only one sample was evaluated, and at least 30 samples per configuration (iOS/Model) had to be present. See text for details.

iOS	Model	ExifIFD	IFD0	IFD1	GPS	MakerNotes
5	iPhone 4	**24**	11	6	9	0
	iPhone 4S	**25**	11	6	9	0
6	iPhone 4	24	11	**6**	9	0
	iPhone 4S	24	11	**6**	9	0
	iPhone 5	24	11	**0**	9	0
7	iPad Air	31	11	6	**9**	6
	iPhone 4	31	11	6	**9**	6
	iPhone 4S	31	11	6	**9**	6
	iPhone 5	31	11	6	**0**	6
8	iPad Air	32	11	6	14	**8**
	iPad Air 2	32	11	6	14	**8**
	iPhone 4S	32	11	6	14	**7**
	iPhone 5	32	11	6	14	**7**
	iPhone 6	32	11	6	14	**10**
9	iPad Air	32	11	6	15	**9**
	iPad Air 2	32	11	6	15	**11**
	iPhone 4S	32	11	6	15	**8**
	iPhone 5	32	11	6	15	**8**
	iPhone 6	32	11	6	15	**11**
	iPhone SE	32	11	6	15	**11**
10	iPad Air	32	11	6	15	**9**
	iPad Air 2	32	11	6	15	**11**
	iPhone 5	32	11	6	15	**8**
	iPhone 6	32	11	6	15	**11**
	iPhone 7	32	11	6	15	**14**
	iPhone SE	32	11	6	15	**11**
11	iPhone 6	32	11	**6**	15	**13**
	iPhone 7	32	11	**0**	15	**18**
	iPhone 8	32	11	**0**	15	**18**
	iPhone SE	32	11	**6**	15	**14**
	iPhone X	32	11	**0**	15	**18**
12	iPhone X	32	11	0	15	22

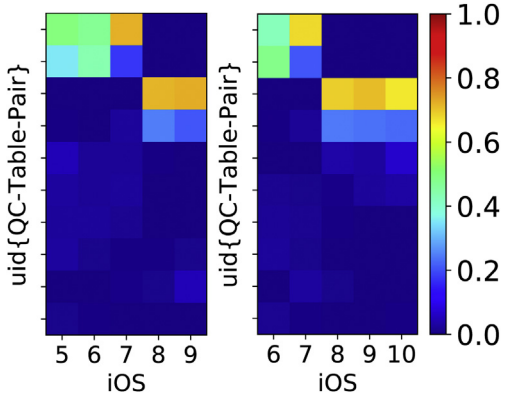

Fig. 6. Distribution of pairs of QC-Tables over iOS-Versions. Left: iPhone 4S, right: iPhone 5.

metadata. Regular digital cameras do not undergo such variations. The change in metadata correlates with the deployment of a new major version of the operating system. Image metadata is changed more often than compression matrices. In fact, we only found one major change in the compression matrices, namely between iOS 7 and iOS 8. Different hardware platforms that run the same operating system appear to contain (up to small differences) the same metadata information.

Classification of hardware models

The findings from the previous section indicate that determining the smartphone hardware model from the header information is much more difficult than for traditional digital cameras. However, to establish a numerical baseline for source identification, we examine this task.

To this end, we train a random forest classifier to identify seven different iPhone models, ranging from the relatively old iPhone 4S, marketed in 2011, to the iPhone X, marketed in 2017. The dataset is randomly subsampled to contain an identical number of smartphone models for each class, and to have each feature be at least contained 30 times in each subclass, which leads to a total of 236 images per model. The resulting dataset is split into 90% training data and 10% test data.

Table 2 shows the confusion matrix for this experiment, where the hypotheses of the classification are along the rows and the true labels are in the columns. An accuracy of 61.9% is achieved. The main diagonal is highly populated which shows correct classifications. First and second minor diagonals also show relatively high

Table 2
Classification of iPhone Models via EXIF directories. The overall accuracy is 0.61933.

iPhone model	4S	5	5s	6	6s	7 Plus	X
4S	119	54	3	1	2	0	0
5	111	169	12	4	1	0	0
5s	4	10	73	51	12	1	0
6	1	0	70	99	49	2	0
6s	0	1	73	80	163	2	1
7 Plus	1	2	3	2	10	184	17
X	0	0	2	0	0	48	218

Table 3
Classification of iPhone Models via quantization matrices. The overall accuracy is 0.35402.

iPhone model	4S	5	5s	6	6s	7 Plus	X
4S	137	89	16	3	1	1	5
5	8	14	9	7	2	6	4
5s	0	3	3	2	1	0	2
6	49	74	137	147	9	25	47
6s	42	54	73	72	212	162	97
7 Plus	0	1	0	0	2	2	2
X	6	7	3	11	15	46	84

Table 4
Classification of iPhone Model via EXIF directories and quantization matrices. The overall accuracy is 0.65653.

iPhone model	4S	5	5s	6	6s	7 Plus	X
4S	118	52	5	2	0	1	0
5	111	172	9	5	0	0	0
5s	3	7	99	96	10	0	0
6	2	2	68	97	18	1	0
6s	0	2	52	34	204	9	3
7 Plus	1	0	2	1	3	178	20
X	0	0	0	0	0	46	212

Table 5
Classification of iOS versions via EXIF directories. The overall accuracy is 0.80409.

iOS	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0
4.0	166	11	11	0	0	0	0	0
5.0	0	95	19	0	0	0	0	0
6.0	5	64	141	0	0	0	0	0
7.0	0	1	0	163	2	1	1	0
8.0	0	0	0	5	151	4	3	1
9.0	0	0	0	3	6	120	50	7
10.0	0	0	0	0	9	34	112	11
11.0	0	0	0	0	3	12	5	152

Table 6
Classification of iOS version via quantization matrices. The overall accuracy is 0.33742.

iOS	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0
4.0	156	111	91	139	8	3	5	5
5.0	5	2	3	1	2	0	0	0
6.0	11	48	65	20	2	1	0	1
7.0	0	2	1	7	4	2	0	0
8.0	1	9	13	5	149	132	87	94
9.0	0	1	0	0	0	0	0	0
10.0	0	0	0	0	0	28	68	53
11.0	0	0	0	1	8	7	13	20

Table 7
Classification of iOS version via EXIF directories and quantization matrices. The overall accuracy is 0.81710.

iOS	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0
4.0	158	8	11	0	0	0	0	0
5.0	4	91	23	0	0	0	0	0
6.0	6	69	134	0	0	0	0	0
7.0	0	0	0	165	1	0	0	0
8.0	0	0	0	3	157	5	1	2
9.0	0	0	0	0	2	117	37	5
10.0	0	0	0	0	4	34	120	4
11.0	0	0	0	0	4	13	10	157

values. This indicates that the classification of iPhone models based on metadata directories mainly confuses adjacent models.

The same experiment for predicting models is performed with the quantization tables as input features. The remaining experimental setup is identical to the previous experiment. In this case, 242 samples per class are obtained. The results are shown in Table 3. The accuracy is considerably lower than for the classification with metadata, at 35.4%. The random forest decides for one of the two adjacent models “iPhone 6” or “iPhone 6s” in approximately two thirds of the cases, which is a major contributor to the low accuracy. Nevertheless, the accuracy is still above random guessing, which is 14.3% for 7 equally distributed classes.

In a third experiment, we merge metadata and quantization features into one long feature vector. Table 4 shows the results for classification with random forest. We obtain 235 images per class, the remaining experimental setup is identical to the previous experiment. The accuracy is 65.6%, which is the better than the results for the individual feature vectors. Combining both features lead to a reduction in the confusion on the second minor diagonal.

Classification of operating systems

Since variations in image signatures are more related to the operating system on Apple than on the hardware model, we predict now operating systems from header information.

The experimental protocol is identical to the previous experiments. The only difference is that the dataset is balanced on the iOS version, which leads to 171 images per class. The data corpus includes all versions from iOS 4 (released in 2010) to iOS 11 (released in 2017). First, only features from the metadata directories are used. These results are shown in Table 5. The accuracy is with 80.4% higher than any predictions for models. Confusions with other iOS versions mainly occur between version 5 and 6, and between

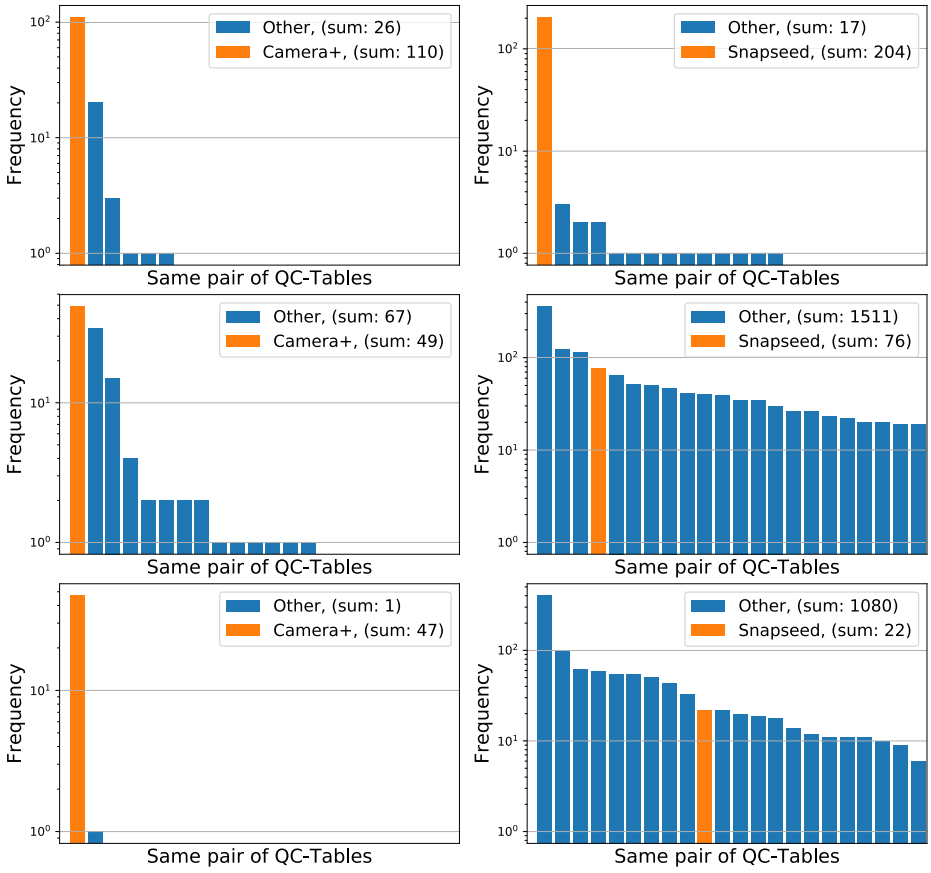

Fig. 7. App identification via quantization matrices. From top to bottom, common quantization matrices are shown that co-occurred with the EXIF:Software tag set to Camera+ (left) and Snapseed (right). In three of the six cases, these quantization matrices were fairly unique to the respective app (orange) and only rarely found in images with other EXIF:Software tags (blue).

versions 9 and 10.

The same experiment is repeated when using quantization tables as features for predicting the iOS version. Interestingly, this task is apparently quite difficult, with an accuracy of only 33.7%. Most of the confusions occur between neighbored versions. The change of quantization tables between iOS 7 and iOS 8 can be clearly seen from the confusion matrix in Table 6.

Combining features from metadata and compression parameters to predict the iOS Version yields the highest accuracy with 81.7%. In Table 7 the main diagonal is populated with large numbers. Misclassifications happen only in very old or new iOS versions.

Uniqueness of smartphone apps

Several apps left an entry in the "EXIF:Software" field, which allows to investigate the case how well it is possible to fingerprint specific apps.

To perform this experiment, we collect all quantization matrices that are found for one specific app and sort them by their relative frequency. In a second step, we search for the same quantization matrices in images where another EXIF:Software tag is set. That way, it becomes possible to quantify how well one of the app's quantization matrices can be used to fingerprint the app.

Fig. 7 shows results for the apps Camera + on the left, and Snapseed on the right. In the two columns are three commonly found quantization matrices shown that co-occurred with these apps. In orange, we show the number of images that we found for the respective app, and in blue, we show images from different apps that share the same quantization matrices. It can be seen, that for Camera+ (left), the first and third quantization matrices are fairly unique. For Snapseed, the first quantization matrix is fairly unique. These are two examples for apps that can be well fingerprinted solemnly from the quantization matrices.

Fig. 8 shows two failure cases of this experiment, here for Instagram (left) and Flickr (right). In both cases, the used quantization matrices are fairly indistinct compared to the quantization matrices of other apps, such that no reliable fingerprint can be established.

Conclusions

We investigated possibilities of associating JPEG header information with images from contemporary smartphones. In contrast to prior studies on images from traditional digital cameras, smartphones are rich software platforms that are regularly modified by software updates. In addition to that, the smartphone user has the choice among many apps for acquiring images.

Both factors together modify the objective of header fingerprinting of image data: the actual hardware platform loses its significance, and instead the fingerprint points towards the software version of the device, or the app that was used to capture the image.

We demonstrate this in four steps. We present a dataset collected from Flickr that consists of more than 200 000 images. On this dataset, we first show that EXIF metadata of smartphones changes over time on Apple devices. We then show that this change is less connected to the actual Apple hardware, but more closely connected to the change of version in the iPhone operating system

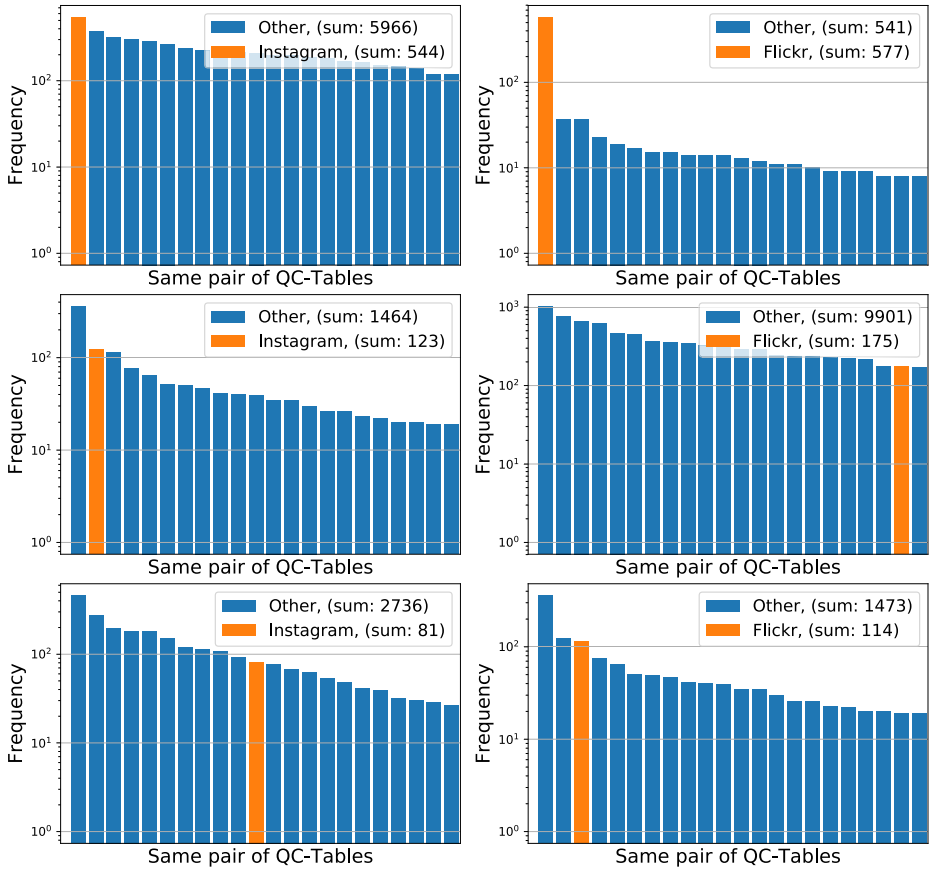

Fig. 8. App identification via quantization matrices. From top to bottom, common quantization matrices are shown that co-occurred with the EXIF:Software tag set to Instagram (left) and Flickr (right). In the shown cases, the app's quantization matrices (orange) are not reliably distinguishable from to images with the same quantization matrix but other EXIF:Software tags (blue).

iOS. These findings are underlined by training automated classifiers, which show to be much more successful in determining iOS versions than hardware platforms. In addition to that, we also show on the concrete example of Camera+ and Snapseed, that some apps use JPEG quantization matrices that are fairly unique to these apps, and that can hence potentially be used to link an image to a specific source app.

Acknowledgement

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

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.
Caldelli, R., Becarelli, R., Amerini, I., 2017. Image origin classification based on social network provenance. IEEE Trans. Inf. Forensics Secur. 12 (6), 1299–1308.
Castiglione, A., Cattaneo, G., De Santis, A., 2011. A forensic analysis of images on online social networks. In: Third International Conference on Intelligent Networking and Collaborative Systems, pp. 679–684.
Electronics, J., Association, I.T.I., 2016. JEITA CP-3451, Exchangeable Image File Format for Digital Still Cameras: Exif Version 2.31 (translated).
Farid, H., 2016. Photo Forensics. The MIT Press.
Giudice, O., Paratore, A., Moltisanti, M., Battiato, S., 2017. A classification engine for image ballistics of social data. In: International Conference on Image Analysis and Processing, pp. 625–636.
Gloe, T., 2012. Forensic analysis of ordered data structures on the example of jpeg files. In: 2012 IEEE International Workshop on Information Forensics and Security, pp. 139–144.
Gloe, T., Kirchner, M., Riess, C., 2013. How We Learned to Stop Worrying about Content and Love the Metadata. IFS-TC Image Forensics Challenge Special Session during WIFS 2013, Guangzhou, China available at: http://ws.binghamton.edu/kirchner/papers/2013_IFC_summary.pdf.
Goljan, M., Fridrich, J., 2008. Camera identification from cropped and scaled images. In: Security, Forensics, Steganography, and Watermarking of Multimedia Contents X. International Society for Optics and Photonics, p. 68190E.
Goljan, M., Fridrich, J., Filler, T., 2010. Managing a large database of camera fingerprints. In: Media Forensics and Security II. International Society for Optics and Photonics, p. 754108.
Hamilton, E., 2004. JPEG File Interchange Format.
Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning, vol. 2. Springer series in statistics New York, NY, USA.
Kee, E., Johnson, M.K., Farid, H., 2011. Digital image authentication from jpeg headers. IEEE Trans. Inf. Forensics Secur. 6 (3), 1066–1075.
Kirchner, M., Gloe, T., 2015. Forensic camera model identification. In: Ho, A.T., Li, S. (Eds.), Handbook of Digital Forensics of Multimedia Data and Devices. John Wiley & Sons, pp. 329–374 (chapter 9).
Li, C., 2010. Source camera identification using enhanced sensor pattern noise. IEEE Trans. Inf. Forensics Secur. 5, 280–287. https://doi.org/10.1109/TIFS.2010.2046268.
Luka, J., Fridrich, J., Goljan, M., 2006. Digital camera identification from sensor pattern noise. IEEE Trans. Inf. Forensics Secur. 1 (2), 205–214.
Ng, N.K.S., 2013. Cell Phone Images in Social Media: an Analysis of Cell Phone Image Structure before and after Social Media Compression. Master's Thesis. Middle Tennessee State University.
Pennebaker, W., Mitchell, J., 1992. JPEG: Still Image Data Compression Standard. Chapman & Hall digital multimedia standards series, Springer US.
Redi, J.A., Taktak, W., Dugelay, J.L., 2011. Digital image forensics: a booklet for beginners. Multimed. Tool. Appl. 51 (1), 133–162.
Salomon, D., 2007. Data Compression: the Complete Reference, 4 ed. Springer Science & Business Media.
Stamm, M.C., Min Wu, M., Liu, K.J.R., 2013. Information forensics: an overview of the first decade. IEEE Access 1, 167–200. https://doi.org/10.1109/ACCESS.2013.2260814.
Wallace, G.K., 1992. The jpeg still picture compression standard. IEEE Trans. Consum. Electron. 38 xviii–xxxiv.