Camera Model Identification Framework Using An Ensemble of Demosaicing Features

Chen Chen
Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA 19104
Email: chen.chen3359@drexel.edu

Matthew C. Stamm
Department of Electrical and Computer Engineering
Drexel University
Philadelphia, Pennsylvania 19104
Email: mstamm@coe.drexel.edu

***Abstract*—Existing approaches to camera model identification frequently operate by building a parametric model of a camera component, then using an estimate of these model parameters to identify the source camera model. Since many components in a camera's processing pipeline are both complex and nonlinear, it is often very difficult to build these parametric models or improve their accuracy. In this paper, we propose a new framework for identifying the model of an image's source camera. Our framework builds a rich model of a camera's demosaicing algorithm inspired by Fridrich et al.'s recent work on rich models for steganalysis. We present experimental results showing that our framework can identify the correct make and model of an image's source camera with an average accuracy of 99.2%.**

I. Introduction

Blindly determining the make and model of an image's source camera is an important forensic problem. Information about an image's source can be used to verify its authenticity and origin. This is particularly important since digital images are often used as evidence during criminal investigations and as intelligence in military and defense scenarios. Additionally, source camera identification techniques can be used to uncover similarities between different camera's internal processing, thus potentially exposing intellectual property theft [1].

Many forensic techniques have been proposed to perform camera model identification [2]. Though some of these perform identification using a set of heuristically designed features [3], the majority operate by building a parametric model of a camera component or the artifacts it leaves behind, then using an estimate of these model parameters to identify the source camera model. Techniques have been proposed to identify the model of an image's source camera using models of a camera's demosaicing algorithm [1], [4], imaging sensor noise [5], lens induced chromatic aberration [6], [7], and proprietary implementations of JPEG compression [8].

Much of the research aimed at increasing the performance of these techniques' performance has focused on improving these parametric models. Most components in a camera's processing pipeline, however, are complex and highly nonlinear. This makes it extremely difficult, if not impossible, to build parametric models that accurately capture intricate characteristics of these components.

Recently, Fridrich et al. developed a novel method of dealing with a similar problem in steganography. Rather than attempting to build accurate parametric models of cover and stego images, Fridrich et al. proposed building a rich model by grouping together a diverse set of simple submodels [9].

In this paper, we propose a new framework for identifying the model of an image's source camera. Our framework builds a rich model of a camera's demosaicing algorithm by grouping together a set of submodels. Each submodel is a non-parametric model designed to capture partial information of the demosaicing algorithm. By enforcing diversity among these submodels, we form a comprehensive representation of a camera's demosaicing algorithm. We then build an ensemble classifier trained on the information gathered by each submodel to identify the model of an image's source camera. We demonstrate the effectiveness of our framework through a set of camera model identification experiments performed on a large database of images.

II. Background

When a digital camera captures an image, light reflected from a real-world scene passes through the camera's lens and optical filter before hitting the imaging sensor. Since most cameras are equipped with only one sensor, they cannot simultaneously record all three primary colors of light at each pixel location. To solve this dilemma, most commercial cameras place a color filter array (CFA) immediately before the sensor. The CFA allows only one color component of light to pass through it at each position before reaching the sensor. As a result, the sensor records only one color value at each pixel location.

Next, the two unobserved color values at each pixel location must be interpolated using a process known as demosaicing. There are generally two types of demosaicing algorithms: non-adaptive and adaptive. Non-adaptive demosaicing algorithms apply a uniform strategy to interpolate unobserved colors throughout the whole image. Most of modern cameras,

Research was sponsored by the U.S. Army Research Office and the Defense Forensics and Biometrics Agency and was accomplished under Cooperative Agreement Number W911NF-15-2-0013. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office, DFBA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

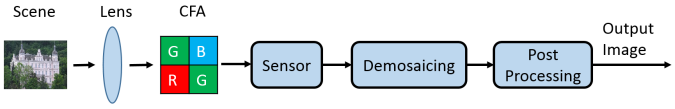

Fig. 1. The processing pipeline in a digital camera.

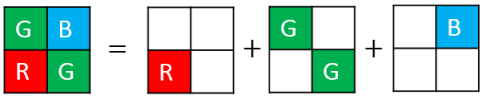

Fig. 2. The Bayer pattern.

however, employ adaptive demosaicing algorithms which can provide higher picture quality. In order to prevent blurring artifacts in textured regions, adaptive algorithms interpolate missing colors in a manner that varies according to the image content. They may also adopt different strategies in different color channels, or interpolate one color channel using the pixel values of other channels. This will introduce complex intra-channel and inter-channel dependencies, making the demosaicing algorithm very nonlinear.

After demosaicing, the image often undergoes a set of post-processing operations such as white balancing, gamma correction, and JPEG compression. A complete overview of a camera's image processing pipeline is shown in Fig. 1.

Though the processing pipeline of virtually all digital cameras are composed of the same components, the implementation of each component typically varies from manufacturer to manufacturer, and from model to model. Furthermore, many of these components leave behind traces in an output image. As a result, many forensic techniques have been developed that use these traces to identify the make and model of an image's source camera. Most of these techniques roughly operate by developing a parametric model of a specific component, or the trace it leaves behind. Next, these parameters are estimated for each image on a large training database of images captured by a variety of different camera models. Finally, these parameter values are used as features to train a classifier to identify an image's source camera.

A significant amount of previous camera model identification research has focused on two components: the CFA pattern and demosaicing algorithm. In [1], Swaminathan et al. jointly estimate the CFA pattern and the demosaicing filter /coefficients, then use these coefficients as features to train a support vector machine to determine the source camera model. To do this, they assume that the color interpolation algorithms are local linear in different textural regions. In their algorithm, images are divided into three different regions according to the gradients, and the color interpolation parameters of each region are estimated linearly. The CFA pattern is determined at last by choosing the candidate CFA pattern that yeilds the smallest re-interpolation error. Cao and Kot develop a partial second-order derivative correlation model to formulate the demosaicing process [4]. They divide all demosaiced color components into 16 categories and build a set of linear demosaicing equations from the partial derivative correlation model for every category. An expectation-maximization reverse classification (EMRC) algorithm is applied to estimate the demosaicing weights for each category. Finally, estimated weights, error statistics and category sizes are used as features for classification.

While both of these algorithms can achieve good performance, they are limited by the fact that both essentially utilize linear or local linear parametric model of the demosaicing process. As we mentioned above, modern demosaicing algorithms are both non-linear and adaptive, and contain complexities that are difficult to capture using these linear models. Furthermore, these complexities may be difficult or impossible to accurately represent even using sophisticated parametric models. While this poses a difficult challenge for camera model identification research, it does not mean that these deomsaicing algorithms cannot be accurately represented. In the next section, we propose a new method to accurately capture the effects of demosaicing algorithms for camera model identification.

III. Source Camera Identification Framework

In our proposed framework, we avoid building parametric models and estimating model coefficients because it's difficult to accurately approximate the components in real cameras. We use another way to represent the CFA pattern and color interpolation algorithm inspired by Fridrich and Kodovsky's work in [9].

Fridrich and Kodovsky developed an universal strategy for steganalysis. They first predict pixel values based on the neighboring pixels with various prediction filters. Under the assumption that natural images are smooth and noise in images is independent of content, the prediction error of stego images which contains hidden information embedded by some steganography techniques will present a different statistical characteristic compared to the error of authentic images. They design a set of diverse submodels to represent the joint probability distribution for each type of prediction error. Each submodel can capture a slightly different trace left by embedding algorithms. Hence, the rich model consisting of diverse submodels can provide powerful information about the embedding algorithms.

We adopt this approach for our source camera identification problem. We build a rich model of demosaicing algorithm used in a camera by generating a diverse set of submodels. Each submodel can only capture part of information about the demosaicing algorithm. To obtain an overall picture of the demosaicing algorithm of a camera, we enforce the diversity of submodels by designing different ways to generate them so that every submodel conveys different aspects of demosaicing information. Thus with a large number of diverse submodels, the rich model grouped from them can yield a much more comprehensive representation of the sophisticated, non-linear color interpolation strategy in cameras compared to mathematical parametric models. We then form our feature space by merging all submodels together and feed them into a multi-class ensemble classifier for camera model identification.

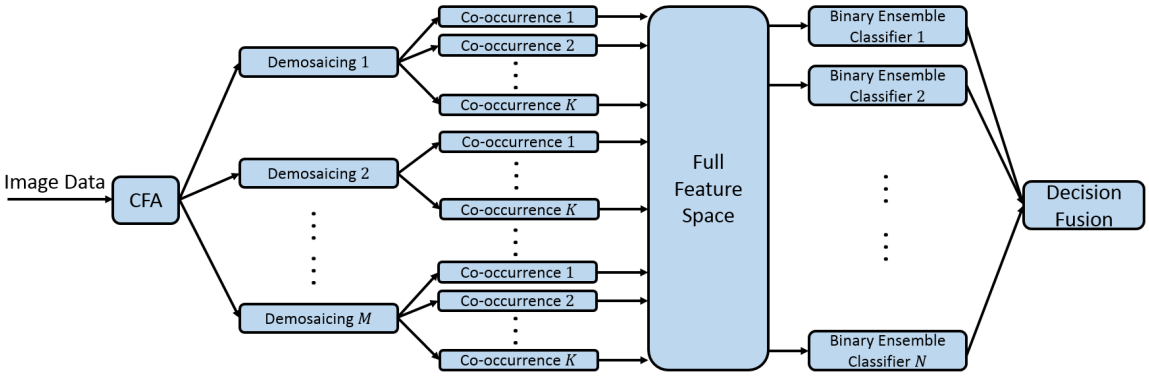

Fig. 3. Architecture of our camera model identification framework.

As is shown in Fig. 3, we reconstruct image data of a camera by re-sampling color components according to the CFA pattern and re-interpolating missing colors with M pre-selected baseline demosaicing algorithms. We then end up with M demosaicing errors which are the differences between the original images and our reconstructed versions. For every demosaicing error, we design K different geometric structures and build submodels by calculating co-occurrence matrices over each geometric structure. The co-occurrence matrix is essentially a way to represent the joint probability distribution of demosaicing errors. A more detailed description of co-occurrence matrix is presented later in this paper. The geometric structure of the joint error distribution can take both different color layers and relative positions in the assumed CFA pattern into account. Thus the $M \times K$ submodels formed from K geometric structures and M baseline demosaicing errors can capture both intra-channel and cross channel correlation of camera's demosaicing algorithm. We now describe the details of how we implement our proposed framework.

A. Feature Collection

1) Re-sampling and Re-interpolation: Among all CFA patterns, the Bayer pattern is the most commonly used. In this paper, we assume all the cameras we inspect employ the Bayer pattern shown in Fig. 2. Since only one color is recorded at one pixel, the blank blocks denote the missing colors in every channel which have to be interpolated. For a particular image $\mathbf{X} = \{\mathbf{R}, \mathbf{G}, \mathbf{B}\}$ where $\mathbf{R}, \mathbf{G}, \mathbf{B}$ represent red, green and blue channel of $\mathbf{X}$ respectively, we apply demosaicing process $Demos$ with re-sampling CFA pattern denoted as CFA and baseline interpolation algorithm H. Then the demosaicing error $\mathbf{E}$ is obtained as follows.

$$\mathbf{E} = \mathbf{X} - \mathrm{Demos}_{CFA,H}(\mathbf{X}) \tag{1}$$

In the demosaicing process, we re-sample the pixel values in three channels according to the chosen CFA pattern. The re-sampled pixel values are supposed to be directly captured by the sensor and have no errors. Then, the other missing two color components have to be domosaiced with nearest neighbor interpolation, bilinear interpolation or other content-adaptive demosaicing algorithms. The error is calculated as the difference between reconstructed image and original one.

2) Quantization and Truncation: After demosaicing with various baseline demosaicing algorithms, we get a set of different demosaicing errors. Every error is a three-layer matrix and pixel values in the positions which should be directly observed by the Bayer pattern are all zeros. If we want to use co-occurrence matrices to approximate its empirical probability distribution, we have to control the value and range of the error. Therefore, we quantize it with step q and truncate it with threshold T.

$$\mathbf{E} \leftarrow \mathrm{trunc}_T\left(\mathrm{round}\left(\frac{\mathbf{E}}{q}\right)\right) \tag{2}$$

We choose $q = 2$ and $T = 3$ to build co-occurrence matrices in our experiment. However, the quantization step and truncation threshold are not unchangeable. In fact, we believe that by closely examining the distribution of demosaicing errors, there should be more efficient and adaptive way to decide their values.

3) Co-occurrence Matrix: We want to find the statistical property of demosaicing errors by building co-occurrence matrices (i.e. each submodel) which are an approximation of the joint probability distribution of error values on designed geometric patterns. The reason why we don't treat error value in every position independently is that there are dependencies not only among errors on different positions within one channel but also among errors in different channels due to the adaptive property of real demosaicing algorithms. Co-occurrence matrices built within and between color channels can capture these intra-channel and inter-channel dependencies which tell us information about demosaicing strategy in cameras.

The format of our demosaicing errors provide us with flexibility to build co-occurrence matrices. Ignoring pixels in every channel which are directly observed according to the CFA pattern, we can design a lot of geometric structures to calculate diverse co-occurrence matrices from demosaiced pixel values. Each co-occurrence matrix conveys its own part of statistical characteristics of the demosaicing error. Here we show an example of generating co-occurrence matrix within

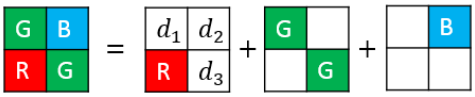

Fig. 4. An example of geometric structure to build co-occurrence matrix of red channel.

red channel in (3) and (4) where $\mathcal{G}1$, $\mathcal{B}$, $\mathcal{R}$ and $\mathcal{G}2$ are the sets of pixel locations which supposed to be directly observed by the Bayer pattern. $\mathbf{C}^{(R)}_{CFA,H}(d_1,d_2,d_3)$ denotes the co-occurrence of red channel with CFA and H as the assumed CFA pattern and demosaicing algorithm. $|\cdot|$ is the cardinality of a set and $\mathbb{1}(\cdot)$ is the indicator function. We count frequency of the triple (d_1,d_2,d_3) appearing in the geometric format shown in Fig. 4 as the joint probability distribution of the three demosaiced values of red channel within the Bayer pattern.

$$\begin{aligned}\mathcal{G}1 &= \{(i,j)|i\ odd, j\ odd\}\\ \mathcal{B} &= \{(i,j)|i\ odd, j\ even\}\\ \mathcal{R} &= \{(i,j)|i\ even, j\ odd\}\\ \mathcal{G}2 &= \{(i,j)|i\ even, j\ even\}\end{aligned} \tag{3}$$

$$\mathbf{C}^{(R)}_{CFA,H}(d_1,d_2,d_3) = \frac{1}{|\mathcal{G}1|}\sum_{(i,j)\in\mathcal{G}1}\mathbb{1}\Big((\mathbf{R}_{i,j},\mathbf{R}_{i,j+1},\mathbf{R}_{i+1,j+1})=(d_1,d_2,d_3)\Big) \tag{4}$$

To capture the inter-channel correlation of demosaicing algorithm, we can design co-occurrence between red and green channel like (5). We calculate two co-occurrence matrices according to the upper and lower geometric structures in Fig. 5 and add them together as the final co-occurrence for red-green channel.

$$\begin{aligned}\mathbf{C}^{(RG)}_{CFA,H}(d_1,d_2,d_3) = &\frac{1}{|\mathcal{G}1|}\sum_{(i,j)\in\mathcal{G}1}\mathbb{1}\Big((\mathbf{R}_{i,j},\mathbf{R}_{i,j+1},\mathbf{G}_{i,j+1})=(d_1,d_2,d_3)\Big)\\ &+\frac{1}{|\mathcal{G}2|}\sum_{(i,j)\in\mathcal{G}2}\mathbb{1}\Big((\mathbf{R}_{i,j},\mathbf{R}_{i-1,j},\mathbf{G}_{i-1,j})=(d_1,d_2,d_3)\Big)\end{aligned} \tag{5}$$

After calculating all designed co-occurrence matrices for all demosaicing errors, we merge (unite) these matrices as the feature set for classification. In our experiment, we only design two geometric patterns to generate co-occurrence matrices because the more co-occurrences we have, the higher dimension of features we get. Due to the curse of dimension in machine learning, overfitting problem easily happens under the circumstances of high dimensional features and insufficient data. Given that it is not feasible gather a tremendous amount of data for every camera model to overcome this problem, dimension of feature is controlled by using a small number of co-occurrences. In the experimental results, we will show that the two designed co-occurrences can still capture an effective amount of demosaicing information to distinguish different camera models.

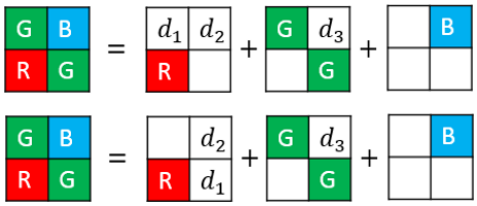

Fig. 5. An example of geometric structure to build co-occurrence matrix of red-green channel.

B. Ensemble Classifier

1) Multi-class Ensemble Classifier: After feature extraction, we merge information from each submodel using a multi-class ensemble classifier. The multi-class classifier is built by grouping a set of binary ensemble classifiers together and using majority voting to fuse all decisions from binary classifers. A more detailed description of these binary ensemble classifiers will be included in Section III-B2. Here we apply all-vs-all strategy [10] to build multi-class classifier from binary classifiers. Suppose we have N camera models to identify, to train a N-class classifier, we have to train a binary ensemble between every possible pair of N camera models and each binary ensemble classifier can only differentiate two camera models which it is trained on. As a result, we end up with a number of $N\times(N-1)/2$ binary ensemble classifiers. Since every binary classifier is an ensemble classifier, our multi-class classifier is essentially an ensemble of ensemble classifiers.

2) Binary Ensemble Classifier: The binary ensemble classifier we use to form our multi-class classifier is a modified version of the classifier proposed by Kodovsky et at. [11]. A diagram illustrating the architecture of each binary ensemble classifier is shown in Fig. 6. The binary ensemble classifier is substantially a random forest with a diverse set of L base learners. Each base learner is a Fisher Linear Discriminant (FLD) classifier whose features are a d_{sub}-dimensional random subspace of the full feature space. To enforce diversity, a different set of d_{sub} feature subset is generated uniformly at random for each base learner. The decision of binary ensemble classifier is reached by performing majority voting amongst the outputs of every base learner.

When training the binary ensemble classifier, we generate different feature subspace of dimension d_{sub} and different bootstrap sample with replacement with roughly 63% unique training samples for each base learner. In Fig. 7, we show the training framework of binary ensemble classifier. For every particular base learner, there are roughly 37% remaining training samples which haven't been used in its training phase and can be exploited to test the trained base learner. After training all base learners and testing on corresponding remaining data, each sample in training data can gather on average $0.37\times L$ votes. Majority voting is applied again to get the decisions for all the training samples. We then can calculate "out-of-bag" (OOB) error of the training data which is an unbiased estimate of the testing error. The two parameters of binary ensemble

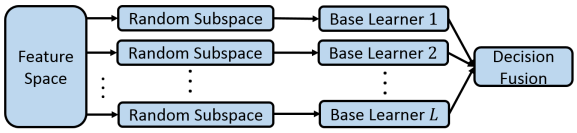

Fig. 6. Flow chart of binary ensemble classifier architecture.

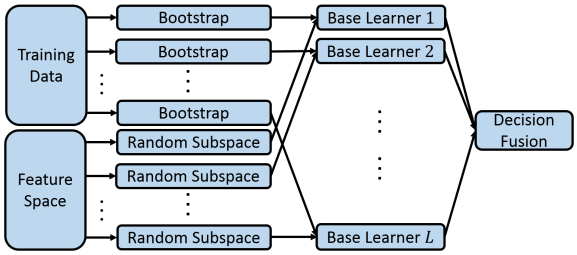

Fig. 7. Flow chart of binary ensemble classifier training.

classifier, optimal dimension of feature subspace d_{sub} and the number of base learners L, are determined through searching algorithms proposed in [11]. The goal of searching is to minimize OOB error. For more detailed algorithms, readers can refer to the original publication.

IV. Simulations and Results

To demonstrate the effectiveness of our proposed camera model identification framework, we constructed an experimental database of images to evaluate its performance. We constructed this database by using 12 different camera models. Each camera was used to capture between 128 and 601 images. This resulted in a set of 3250 full resolution images. Each image was captured and stored as a JPEG using the camera's default settings. A list of the camera models used to collect these images is shown in Table I.

Next, we divided each of these images into a set of 512×512 pixel subimages. To ensure that each subimage was suitable for extracting information about the demosaicing algorithm, we measured the texture, intensity, and flatness of each subimage using the features proposed in [12]. Subimages that were saturated or that contained insufficient texture or illumination were excluded. This is acceptable because in practice, an investigator will only make use of blocks of a full resolution that are suitable for forensic examination. The remaining subimages were used to form an experimental database of 58445 512×512 pixel images, with between 2118 and 10398 images from each camera model.

A. Experiment 1

We used this database to experimentally measure the performance of our camera model identification framework. In this experiment, we randomly chose 90% of the images from each camera to use for training our classifier. The remaining 10% of images from each model were used for testing our trained classifier.

As we discussed in Section III, we wish to capture as much interpolation information as possible by designing a large set of co-occurrence matrices. If too many submodels are used, however, the dimensionality of the feature space may grow too large relative to the amount of training data, and we risk overfitting.

TABLE I
Camera Models Used in Our Experiment

Camera ID	Make	Model
1	Nikon	D7100
2	Canon	PowerShot SX500 IS
3	Canon	PowerShot N2
4	Sony	Alpha 58
5	Sony	A6000
6	Xiaomi	Mi 1S
7	Samsung	Galaxy S4
8	Samsung	Galaxy S5
9	Motorola	Moto X 2013
10	Motorola	Moto X 2014
11	Apple	iPhone 5S
12	Apple	iPhone 6

To avoid overfitting in this experiment, we used four submodels to construct our feature space. These submodels were built using two baseline demosaicing algorithms: nearest neighbor and bilinear interpolation. When computing the demosaicing residual for each of these algorithms, we resampled each image using the Bayer mask. We then formed one intra-channel (red channel) and one inter-channel (red-green channel) co-occurrence matrix from the residuals of each baseline demosaicing algorithm according to (4) and (5).

When constructing the co-occurance matrices, we used a quantization step $q = 2$ and truncation threshold $T = 3$. As a result, the dimension of the 3-D co-occurrence matrix for each submodel was $(2T + 1)^3 = 343$. After merging all co-occurrences together, the dimension of full feature space was $343 \times 4 = 1372$.

After training our classifier, we used it to identify the model of the source camera of each image in the test set. The classification results of this experiment are shown in the confusion matrix in Table II. The percentage of images correctly classified for each model are highlighted in bold.

From these results, we can see that our classifier achieved an average accuracy of 99.2%. The minimum classification accuracy of 97.3% was obtained for Camera 12 (iPhone 6). These results verify the ability of our framework to effectively identify make and model of an image's source camera. Furthermore, we can also see that the most mixture of Camera 12 (iPhone 6) is with Camera 11 (iPhone 5S). This is likely because both of these cameras are made by the same manufacturer, so it is possible that their demosaicing algorithms have aspects in common.

B. Experiment 2

We conducted a second experiment to verify that our classifier is not overfitting to device-specific information. To do this, we gathered a set of images from an iPhone 5S and two iPhone 6's that were not used to create our first experimental database. We then pre-processed these images in the exact same way as the first experiment by dividing them into subimages and excluding overly smooth or saturated blocks. This resulted

TABLE II
CONFUSION MATRIX SHOWING CLASSIFICATION RESULTS OF EXPERIMENT 1

		True Model											
		1	2	3	4	5	6	7	8	9	10	11	12
Identified Model	1	**99.6%**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	2	0.0%	**100%**	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.3%	0.0%	0.0%	0.0%
	3	0.2%	0.0%	**99.8%**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.5%
	4	0.2%	0.0%	0.0%	**99.8%**	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	5	0.0%	0.0%	0.0%	0.0%	**99.8%**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	6	0.0%	0.0%	0.0%	0.0%	0.0%	**99.6%**	0.0%	0.0%	0.3%	0.0%	0.0%	0.0%
	7	0.0%	0.0%	0.0%	0.0%	0.0%	0.3%	**99.5%**	0.4%	0.0%	0.0%	0.0%	0.0%
	8	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.5%	**99.3%**	0.0%	0.4%	0.0%	0.0%
	9	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	**98.6%**	0.4%	0.1%	0.2%
	10	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.4%	0.3%	**98.3%**	0.1%	0.2%
	11	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.3%	0.0%	**98.8%**	1.8%
	12	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.9%	0.8%	**97.3%**

TABLE III
CLASSIFICATION RESULTS OF NEW CAMERA DATA IN EXPERIMENT 2

		True Model		
		11-A	12-A	12-B
Identified Model	1	0.0%	0.0%	0.0%
	2	0.1%	0.3%	0.0%
	3	0.0%	1.6%	0.0%
	4	0.1%	0.0%	0.0%
	5	0.0%	0.0%	0.0%
	6	0.1%	0.0%	0.0%
	7	0.0%	0.0%	0.0%
	8	0.1%	0.2%	0.0%
	9	0.5%	0.4%	0.0%
	10	0.0%	0.5%	0.0%
	11	**99.0%**	0.8%	0.9%
	12	0.3%	**96.3%**	**99.1%**

between 1911 and 3951 512×512 pixel testing images for each new camera.

Next, we used our trained classifier from our first experiment to identify the model of the source camera of each of these new images. The classification results of this experiment are shown in Table III. Entries highlighted in bold correspond to correct classification results.

From these results, we can see that the accuracy of testing on totally new data is consistent with the results presented in the confusion matrix in Table II. This verifies that our framework is not overfitting to characteristics of individual devices. Furthermore, it reinforces that our proposed framework can learn color interpolation information and identify source camera models with high accuracy.

We note that in this experiment, the classification accuracy for the first new iPhone 6 (12-A) decreases slightly to 96.3%. After checking the pre-selected testing subimages from this camera, we found there are a small amount of dark and noisy subimages. When extracting feature from these subimages, noise components may be dominant over color interpolation information and confuse our classifier. This also tells us that our designed pre-selection strategy according to three image features can be improved. In future work, we plan to come up with more effective strategy to ensure the quality of data.

V. CONCLUSION

In this paper, we propose a new framework for performing source camera identification. Instead of building a parametric model of a camera's demosaicing algorithm, we build a rich model from a diverse set of submodels. We compute a set of demosaicing errors, which are the differences between an image and a re-interpolated of it using several baseline demosaicing algorithms. Each submodel is formed as a structured joint distribution of the demosaicing errors (represented as co-occurrence matrix). Each submodel can capture partial information about the demosaicing algorithm in a camera. We then combine all submodels together, and feed them into a multi-class ensemble classifier. We verify the effectiveness of our proposed framework through a series of experiments. Our experimental results show that our framework can identify the correct make and model of an image's source camera with an average accuracy of 99.2%.

REFERENCES

[1] A. Swaminathan, M. Wu, and K. Liu, "Nonintrusive component forensics of visual sensors using output images," *Information Forensics and Security, IEEE Transactions on*, vol. 2, no. 1, pp. 91–106, 2007.

[2] M. C. Stamm, M. Wu, and K. J. R. Liu, "Information forensics: An overview of the first decade," *IEEE Access*, vol. 1, pp. 167–200, 2013.

[3] M. Kharrazi, H. Sencar, and N. Memon, "Blind source camera identification," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, vol. 1, Oct. 2004, pp. 709–712.

[4] H. Cao and A. C. Kot, "Accurate detection of demosaicing regularity for digital image forensics," *Information Forensics and Security, IEEE Transactions on*, vol. 4, no. 4, pp. 899–910, 2009.

[5] T. Filler, J. Fridrich, and M. Goljan, "Using sensor pattern noise for camera model identification," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, Oct. 2008, pp. 1296–1299.

[6] L. T. Van, S. Emmanuel, and M. S. Kankanhalli, "Identifying source cell phone using chromatic aberration," in *Multimedia and Expo, 2007 IEEE International Conference on*. IEEE, 2007, pp. 883–886.

[7] K. San Choi, E. Y. Lam, and K. K. Wong, "Source camera identification using footprints from lens aberration," in *SPIE Electronic Imaging 2006*, 2006, pp. 60 690J–60 690J.

[8] E. Kee, M. Johnson, and H. Farid, "Digital image authentication from jpeg headers," *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 3, pp. 1066–1075, Sep. 2011.

[9] J. Fridrich and J. Kodovskỳ, "Rich models for steganalysis of digital images," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 3, pp. 868–882, 2012.

[10] M. Aly, "Survey on multiclass classification methods," *Neural Netw*, pp. 1–9, 2005.

[11] J. Kodovskỳ, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 2, pp. 432–444, 2012.

[12] M. Chen, J. Fridrich, M. Goljan, and J. Lukáš, "Determining image origin and integrity using sensor noise," *Information Forensics and Security, IEEE Transactions on*, vol. 3, no. 1, pp. 74–90, 2008.