Robustness of CNN-based Camera Model Identification[*]

Jiarong Pan
Friedrich-Alexander Universität Erlangen-Nürnberg
garypan4@gmail.com

21.02.2020

Abstract

In multimedia forensics, camera model identification is an important task for various applications ranging from criminal investigations to fraud detection. Recently, the data-driven forensic methods show state-of-the-art performance in identifying and verifying camera devices. Most of the distinctive traces of the camera model is in high-frequency features, so using a constrained convolutional layer as the first network layer became a popular choice. However, researchers found that most learning-based camera model identification methods are challenged by out-of-distribution data, i.e., the data with unseen features that are not included in the training set. In this situation, it's very difficult to tell how much we can trust the classifier, or how robust are they. Therefore, we investigate the effects on the CNN-based camera model classifier of using images from unseen camera models, post-processed images and images from different devices.

1 Introduction

Multimedia information, such as digital images, is frequently used in numerous important settings, such as evidence in legal proceedings, criminal investigations, and military and defense scenarios. The goal of camera model identification is to determine the make and model of the source camera. By exploiting the subtle artifacts and traces that relate to the camera model, researchers can build learning-based models that learn from this characteristics of cameras. Most of these methods have great performances in experiments. However, one of the disadvantages of learning-based methods is that they are sensitive to out-of-distribution data, i.e., if a test image is too different from the training set images, it's hard for this method to find out these characteristics. For example, there are lots of images in social media which are re-compressed or manipulated, if we try to identify these images using a model trained on other data, which only include limited types of re-compression and manipulations, the result would be affected by the unseen manipulations and re-compression.

In this work, we want to test the robustness of a CNN-based camera model classifier in three different situations: **images from unseen camera models**, **post-processed images** and **images from different devices**. To evaluate the robustness of the classifier, we use the **accuracy of the classifier**, and the **mean confidence of the output**. Note that we use the maximal values of the softmax output as confidence of the prediction. So if the CNN has high mean confidence and low variance, it could be consider that the classifier is confident about it's decision. Otherwise, the classifier is confused by the input.

[*]This is a research project at the IT Security Infrastructures Lab, Friedrich-Alexander Universität Erlangen-Nürnberg. Advisor: Benedikt Lorch, Anatol Maier, Dr. Christian Riess. The code is available on github.com/JoiDvision/CamModel

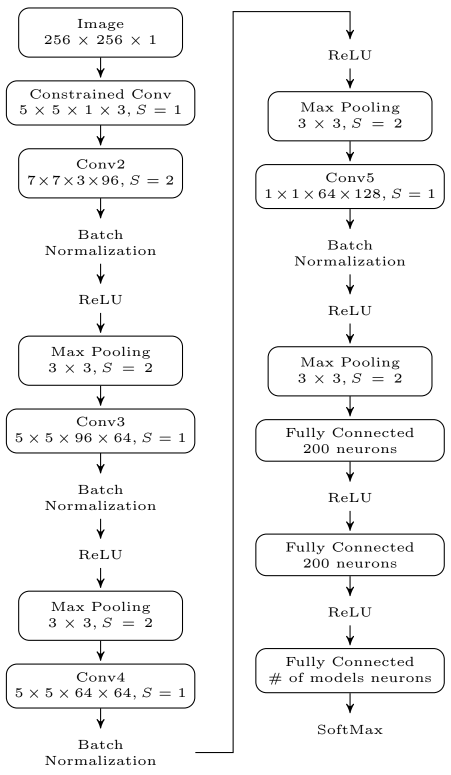

Figure 1: Baseline CNN with constrained convolutional layer as the first layer.

2 Camera model identification

Typically, CNN architectures operating on images are trained to learn discriminant features from the image content. In contrast, our CNN model needs to be capable of learning content invariant features in order to identify camera models. In this work, the CNN-based camera model classifier is based on the work by Bayar and Stamm [2].

The baseline architecture we use is shown in Figure 1. We can see that the baseline architecture consists of one constrained convolutional layer which is followed directly by four convolutional layers and three fully connected layers. All the regular convolutional layers are followed by a batch normalization layer [4], rectified linear unit activation function [5] and a max-pooling layer. The last layer is a softmax activation function and the class of the input image corresponds to the highest activated neuron in the output layer. In this work, the input layer of our CNN is the green channel of an image patch sized 256 × 256 pixels. The constrained convolutional layer extracts the high frequency characteristics from the input images. Then the high-level features of these characteristics will be extracted through the following hidden convolutional layers.

As we mentioned above, common CNN architectures doesn't suppress the content of images and it may correlate the image's content in the training set with the image models, which is unwanted.

The constrained convolutional layer by Bayar and Stamm [1] is to combat this issue, it can not only suppress the image's content, but also adaptively learn pixel values dependencies while training CNN. The pseudocode of the training is given in Algorithm 1.

Algorithm 1 Training algorithm for constrained convolutional layer

```
1: Initilize $a_k$ 's using randomly drawn weights
2: i = 1
3: while $i \leq max_{iter}$ do
4:     Do feedforward pass
5:     Update filter weights through stochastic gradient
6:     descent and backpropagate errors
7:     Set $a_k^{(1)}(0,0) = 0$ for all $K$ filters
8:     Normalize $a_k^{(1)}$ such that $\sum_{l,m\neq 0} a_k^{(1)}(l,m) = 1$
9:     Set $a_k^{(1)}(0,0) = -1$ for all $K$ filters i = i + 1
10:    if training accuracy converges then
11:        exit
12:    end if
13: end while
```

3 Experiment

In this part, we show an experimental study about the impact of different test data on the robustness of the CNN-based classifier. We used the baseline CNN architecture in Figure 1 to adaptively extract image traces left by camera and identify camera models of images.

Data collection & Training

We choose images from the dresden image database [3] as our training set. More specifically, we pick the images from three models: 567 images of **Canon Ixus70**, 752 images of **Nikon D200** and 1040 images of **Olympus mju-1050SW**, in total 2359 images, such that they all share similar image contents.

To train our CNN, we first randomly select 1,887 images from our experimental database. Next, we divide these images into 256 × 256 pixel patches and retain the central 25 patches from the green channel of each image for our training database. In this database, each block corresponds to a new image that has its corresponding camera model label. In total, our training database consists of 47,175 patches.

When training our CNN, we set the batch size equal to 64 and the parameters of the stochastic gradient descent as follows: momentum $\mu = 0.9$, decay $\delta = 0.0005$, and a learning rate $\epsilon = 10^{-3}$ that decreases every 5 epochs by a factor $\gamma = 0.5$. We trained the CNN in each experiment for 45 epochs. To combat the class imbalance problem, we use the class weight w_k for each class k, which is shown in Equation 1.

$$w_k = \frac{\text{total number of images}}{\#\text{ of images of class k} \times 2} \tag{1}$$

Finally, to evaluate the performance of our CNN, we created a testing database by dividing the remaining 472 images that are not used for the training into 256×256 pixel patches in the same manner described above. In total, our testing database consisted of 11,800 patches.

Result

We randomly select 6,400 patches from our testing database, the camera models Canon Ixus70, Nikon D200 and Olympus mju-1050SW have respectively 1,403, 1,938 and 3,059 images. Note that there are more Olympus mju-1050SW images than the others in our testing database. The accuracy of our CNN architecture for each camera model is shown in Table 1, it turns out the model has high overall accuracy up to 99.90%.

Camera models	Accuracy	Number of images
Canon Ixus70	100%	1,403
Nikon D200	99.69%	1,938
Olympus mju-1050SW	100%	3,059
Overall Accuracy	99.90%	

Table 1: Prediction accuracy to test data. The rightmost column shows the number of testing images from the corresponding camera model. Note that the images here are 256 × 256 patches divided from full sized images.

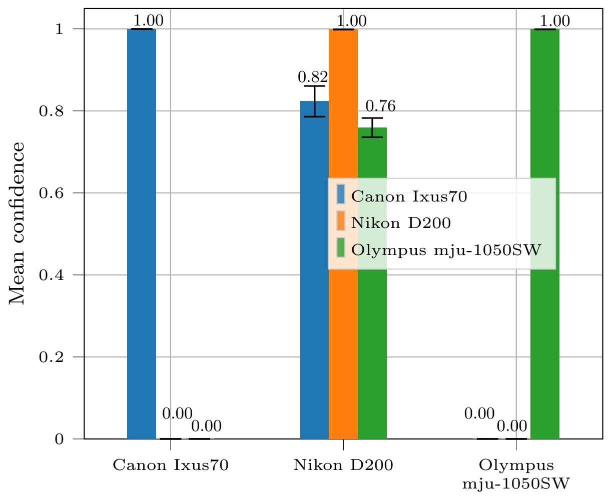

Figure 2: Mean prediction confidence to test data, x axis indicates the images from three testing models, y axis shows the value of mean prediction confidence. The color bars tell us which model the image is classified into.

As we can see, Figure 2 depicts the mean and variance of the confidence for each class. For the classes Canon Ixus70 and Olympus mju-1050SW, our CNN classifies them correctly with confidence of 1. In the case of Nikon D200, it has small number of misclassifications, whose mean confidence are 0.82 for Canon Ixus70 and 0.76 for Olympus mju-1050SW. Considering it has 99.69% accuracy, it could be interpreted that our CNN is confident and makes the correct prediction for most of the cases, but it could still be confused by some exceptions.

3.1 Unseen camera models

This experiment is to test the impact of images from unseen camera models on the robustness of our classifier. Note that in this experiment our classifier classifies the input images from the five testing camera models as one of the three training camera models, as we trained our model on a closed set classification task using the mutual exclusive class labels $\mathcal{C} = \{\text{Canon Ixus70}, \text{Nikon D200}, \text{Olympus mju-1050SW}\}$. Since the output labels are all different from the input labels, the classifier should make decision without high confidence if it is robust to image from unseen camera model. It could be interpreted that the classifier cannot be sure which model the image comes from.

Camera Models	Brand	Contents	Number of images
Agfa DC-830i	different	similar	363
Canon Ixus55	same	similar	224
Canon Powershot64	same	different	188
Nikon D70	same	similar	369
Sony W170	different	different	405
Total number of images			1,549

Table 2: Five unseen camera models with different relations in terms of brand and image content. The rightmost column shows the number of testing images from the corresponding camera model. Note that the images here are full sized images.

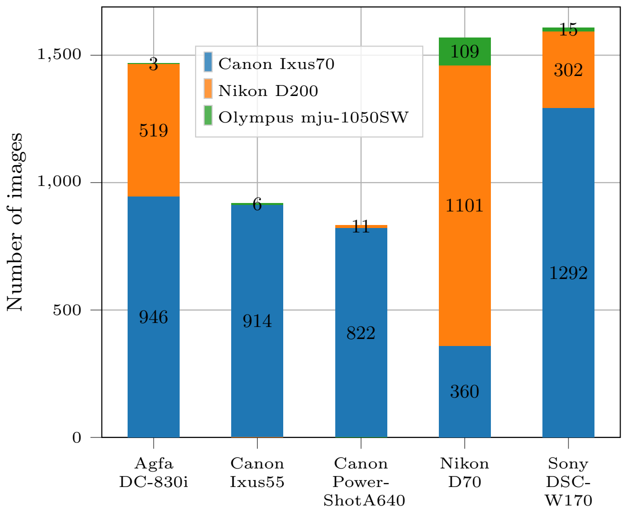

Figure 3: Prediction to five unseen models. The numbers on the bars indicate the number of images that are classified to the corresponding classes.

Data collection

As shown in Table 2, we choose five more models in the the dresden image database, which are **Agfa DC-830i**, **Canon Ixus55**, **Canon PowershotA640**, **Nikon D70** and **Sony W170**. In order to see if the brand and image content affect the result of our classifier, these five models have different relations to the training models in terms of brand and image contents. These 1,549 images are also divided into 256 × 256 patches in the same manner mentioned before. In total, our unseen database has of 38,725 patches.

Result

We randomly selected 6,400 patches from our unseen database and evaluate the classifier's most likely class prediction, as well as its prediction confidence, which is shown in Figure 3 and 4, respectively.

We can notice that for the two Canon cameras, Canon Ixus55 and Canon PowerShotA640, the classifier classifies almost all the images from these two models as Canon Ixus 70, which has the same manufacturer as the two models. The mean prediction confidence of Canon Ixus55 and Canon PowerShotA640 are both equal or higher than 0.91, so it implies that the classifier is confident about the decisions it makes. However, there is difference between the prediction result of these two camera models. For the images from Canon Ixus55, the classifier tends to classify a small number of images as Olympus mju-1050SW, while for images from Canon PowerShotA640, as Nikon D200. In addition, the image contents of Canon Ixus55 are similar to the training set while Canon PowerShotA640 are different. Nevertheless, it seems this difference does not affect the result of these two camera models' images.

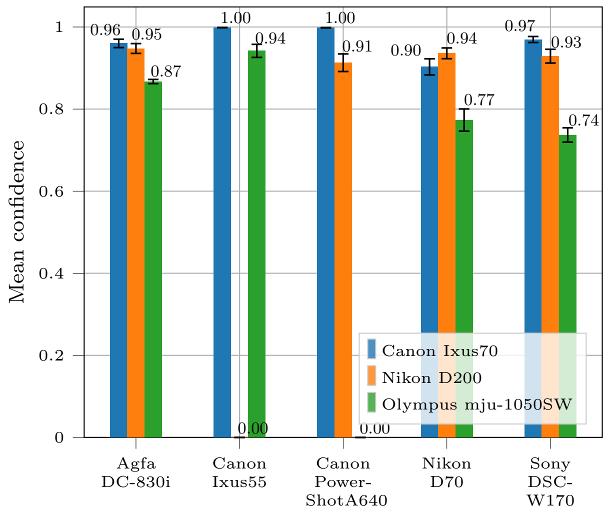

Figure 4: Mean prediction confidence to five unseen camera models.

Similar result is also shown for images from Nikon D70. Our classifier tends to classify them as image from Nikon D200. It could be the reason that Nikon D70 and Nikon D200 both have the same manufacturer. As for the rest 3,077 images from the other two camera models, Agfa DC-830i and Sony DSC-W170, out of these 2,238 are classified as Canon Ixus70, which makes about 72.7%. Besides, when the classifier classifies the images from these three camera models as Canon Ixus70 and Nikon D200, it always has at least 0.90 mean prediction confidence. The mean prediction confidence to Olympus mju-1050SW is also considerable with a value higher than 0.74.

For all images of the unseen test models, the result doesn't show a relation to the image content. The explanation for this is the usage of constrained convolutional layer, which extracts the high frequency characteristics of the camera instead of the scenes and objects of the images.

Summary

In this set of experiment, we have shown that the CNN-based camera model classifier is affected by the manufacturer of the camera models and it is robust to the contents due to the constrained convolutional layer. However, the classifier makes wrong decision always with high confidence. To some extend, this classifier is not robust to unseen camera models, because it doesn't show "hesitation" while making choices.

3.2 Post-processed images

In experiment of **post-processed images**, we perform three different post-processing techniques on the images from the testing set respectively. The post-processing and their parameters are shown in Table 3. The parameters of these operations are referenced from the manipulation detection experiment by Bayar and Stamm [2].

Operations	Parameters
JPEG compression	$QF = 70$
Gaussian Noise	$\sigma = 2$
Gaussian Blur	$\sigma = 1.1$, $K_{size} = 5 \times 5$

Table 3: Operations and parameters of the post-processing experiment. The parameters are referenced from the manipulation detection experiment by Bayar and Stamm [2].

Camera Models	Accuracy	Number of images
Canon Ixus70	58.10%	1,439
Nikon D200	93.21%	1,886
Olympus mju-1050SW	68.16%	3,075
Overall Accuracy	73.16%	

Table 4: Prediction accuracy to JPEG compressed images with quality factor of 70. The images here are 256 × 256 patches divided from full sized images.

3.2.1 JPEG Compression

We perform JPEG compression with a quality factor of 70 on all the patches from our test database. In the evaluation stage, we randomly choose 6,400 patches as input to our CNN.

Result The classification result of JPEG images is shown in the Figure 5. Compared to the accuracy of Table 1, although they are all from the same camera models and also have the same image contents, the overall classification accuracy has dropped significantly after JPEG compression as shown in Table 4.

The accuracy of Olympus mju-1050SW has dropped more than 40%. For Canon Ixus70, it's even more than 50%. Surprisingly, the accuracy of Nikon D200 only decrease by about 6.5%, from 99.69% to 93.21%, which is much better than the other two models.

In terms of the prediction confidence, the mean confidences of the corresponding correct class are still the highest compared to the other two classes. So the classifier is still more certain when making the correct decision. But comparing Figure 6 with Figure 2, it shows that not only the difference between these mean confidences become greatly smaller except for Nikon D200, but also the confidences for the corresponding correct classes are decreased.

Summary The result of this experiment shows that, the classifier is robust to the JPEG compressed images from Nikon D200, but it fails when classifying the JPEG compressed images from the other two models. Considering the mean confidence when classifying Nikon D200's images in this experiment doesn't show great difference from Figure 2, one possible explanation for this robustness is that compared to the high frequency characteristics left by the camera, the artifacts caused by the jpeg compression are in different frequency range, such that the classifier consider these artifacts as unimportant when classifying images from Nikon D200. Nevertheless, when the classifier misclassifies images, it is always less sure about its decision compared to when it correctly classifies.

3.2.2 Additive Gaussian Noise

A Gaussian noise with with zero mean and standard deviation of $\sigma = 2$ is added to the patch in the test database in this experiment. To evaluate the robustness of our classifier to additive Gaussian noise, we randomly select 6,400 patches

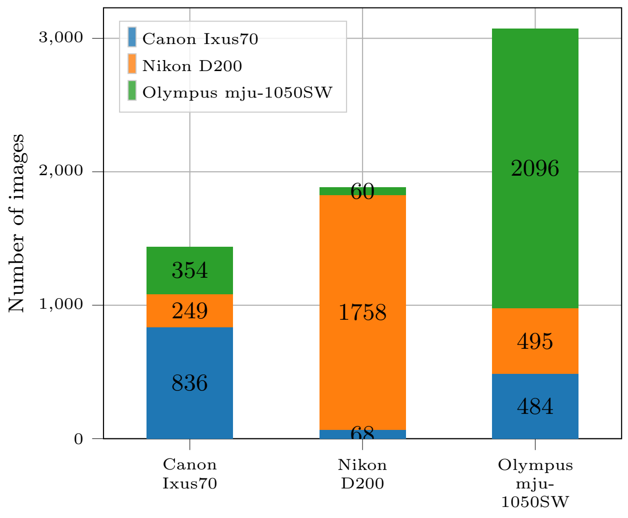

Figure 5: Prediction to JPEG compressed images with quality factor of 70.

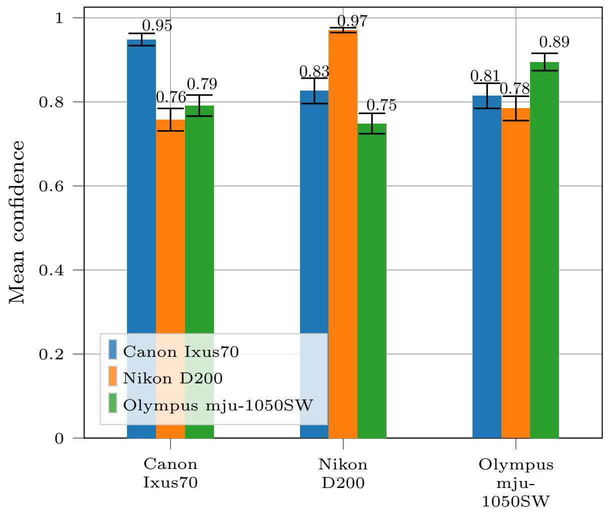

Figure 6: Mean prediction confidence of JPEG compressed images with quality factor of 70.

Camera Models	Accuracy	Number of images
Canon Ixus70	99.49%	1,371
Nikon D200	55.94%	1,977
Olympus mju-1050SW	23.53%	3,052
Overall Accuracy	59,65%	

Table 5: Prediction accuracy to images added Gaussian noise with zero-mean and standard deviation $\sigma = 2$. The images here are 256×256 patches divided from full sized images.

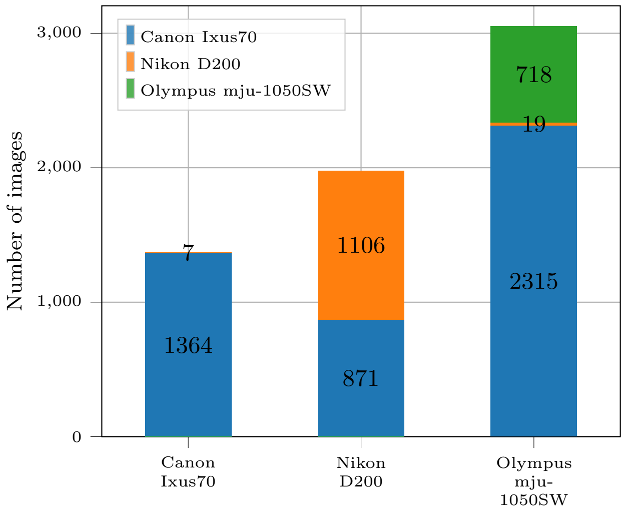

Figure 7: Prediction to images added Gaussian noise with zero-mean and standard deviation $\sigma = 2$.

from the test set and distort each patch with this Gaussian noise.

Result Figure 7 and Table 5 show the classification result for these noisy images. In contrary to the JPEG compression experiment, the accuracy of Canon Ixus70 is nearly the same as in the baseline experiment, while the accuracies of Nikon D200 and Olympus mju-1050SW have downgraded severely, especially Olympus mju-1050SW, it decreases from 100% to 23.53%.

As shown in Figure 8, the prediction confidence for Canon Ixus70 doesn't change much as well. The mean confidence for classifying noisy Canon Ixus70 images as Canon Ixus70 images is still 1, and the 0.77 mean confidence for Nikon D200 is just caused by a very small number of misclassifications. When classifying images from the other two models, Nikon D200 and Olympus mju-1050SW, as Canon Ixus70 images, they both show more than 0.9 confidence to this wrong decision. In the case of Olympus mju-1050SW, the confidence is even higher compared to a correct decision.

Summary In this experiment, it shows that the CNN-based camera model classifier is robust to additive Gaussian noise only for Canon Ixus70. For noisy images from the other two models, it tends to classify the noisy images as they are from Canon Ixus70 with high confidence. It leads to low accuracy for images from these models, especially for images from Olympus mju-1050SW. The explanation for this phenomenon could be that there is similarity between the high frequency characteristics of image from Canon Ixus70 and the Gaussian noise, such that it leads to the classifier tends to classify noisy image as Canon Ixus70's image.

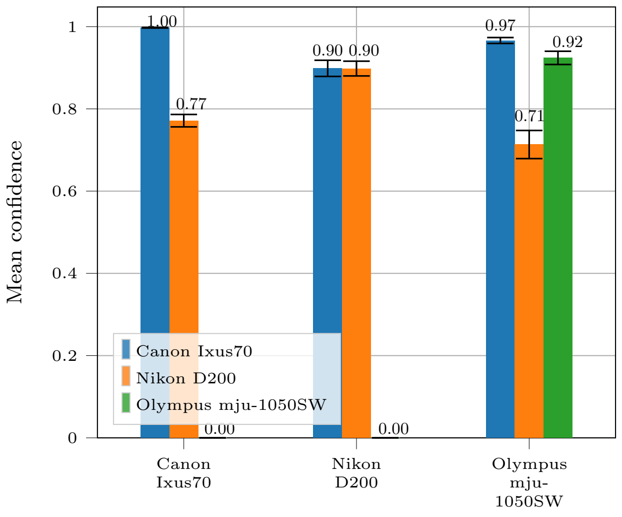

Figure 8: Mean prediction confidence to images added Gaussian noise with zero-mean and standard deviation $\sigma = 2$.

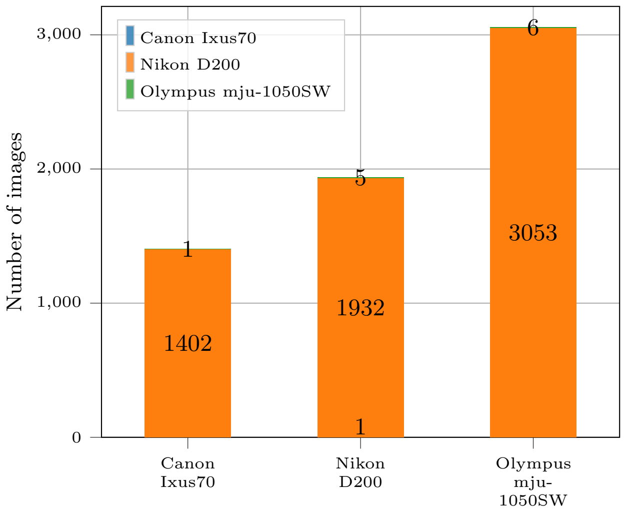

Figure 9: Prediction to images blurred by Gaussian filter with kernel size of 5×5 and standard deviation $\sigma = 1.1$.

3.2.3 Gaussian Blur

A Gaussian blur operation with kernel size of 5×5 and standard deviation $\sigma = 1.1$ is performed on the image patches in the test database. We randomly select 6,400 patches and distort them with this Gaussian filter before using it as input to the classifier.

Result The result of this experiment is shown in Figure 9 and Figure 10. Almost all images from these three models are classified as Nikon D200. In addition, the mean confidence when classifying images as Nikon D200 images are all 1, while for the other two models are low, which are either around 0.6 or 0.

Summary In this section, the results show that the classifier is severely affected by the Gaussian blur operation, which causes it to tend to recognize blurred images as images from Nikon D200. One possible explanation for this could be that the traces left by Nikon D200 has lower frequency compared to the other two images since the Gaussian blur process is a low pass filtering operation.

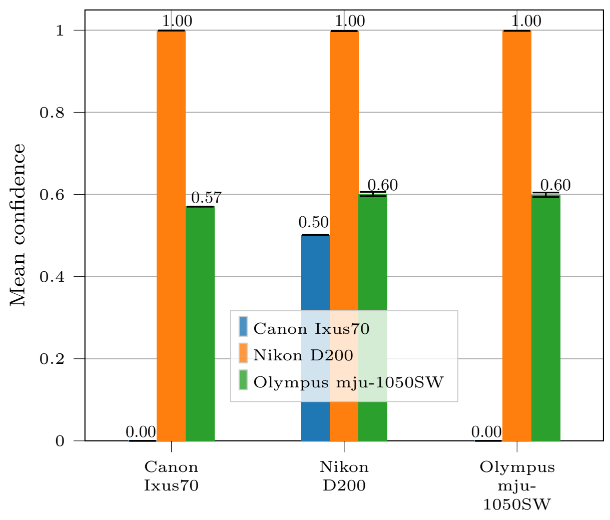

Figure 10: Mean prediction confidence to images blurred by Gaussian filter with kernel size of 5×5 and standard deviation $\sigma = 1.1$.

3.3 Different devices from same model

In order to study the robustness of our CNN-based camera model classifier to the device-dependent characteristics, we separate the images of our training set according to models and devices. Each model has more than one devices, more specifically, three devices for Canon Ixus70, two devices for Nikon D200 and five devices for Olympus mju-1050SW.

Data Collection & Training

We select the images from one device of each model. These three devices from Canon Ixus70, Nikon D200 and Olympus mju-1050SW has 187, 372 and 204 full sized images respectively. Then we divide these images into 256×256 pixels patches and train our CNN with them in the same manner described in Section 3. In total, there are 19,075 patches in this dataset, of which 15,260 images are used for training and 3,815 images for evaluating the performance of our CNN.

The other images, namely 380 full sized images from Canon Ixus70, 380 from Nikon D200 and 836 from Olympus mju-1050SW, are used as the testing data for this experiment. Again, separate them in 256×256 pixels patches. In total, the number of images for testing robustness to device-dependent characteristics is 35,075.

Result

The performance of our CNN in this experiment turns out to be really similar compared to Table 1 and Figure 2. Using the images from different devices as input, the results are depicted in Figure 11 and Figure 12. As we can see, the accuracy for this experiment is at least 98.75%.

In terms of confidence of prediction, the mean confidence for the ground truth labels are about 1, similar to Figure 2. Compared to Canon Ixus70 and Nikon D200, the Olympus mju-1050SW is slightly better as for the prediction confidences for wrong classes are around 0.6, while the other two models have around 0.8 confidences.

Summary

We have demonstrated that the classification accuracy and the prediction confidence didn't change much for different de-

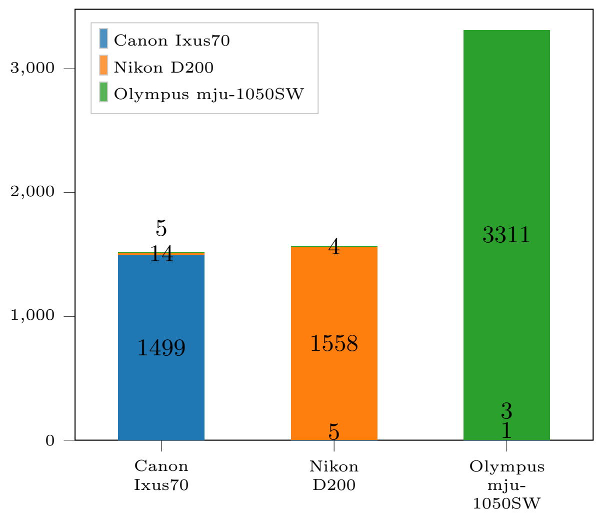

Figure 11: Prediction to images from different devices of same model. The numbers from top to bottom indicate the number of images classified as Olympus mju-1050SW, Nikon D200 and Canon Ixus70 respectively.

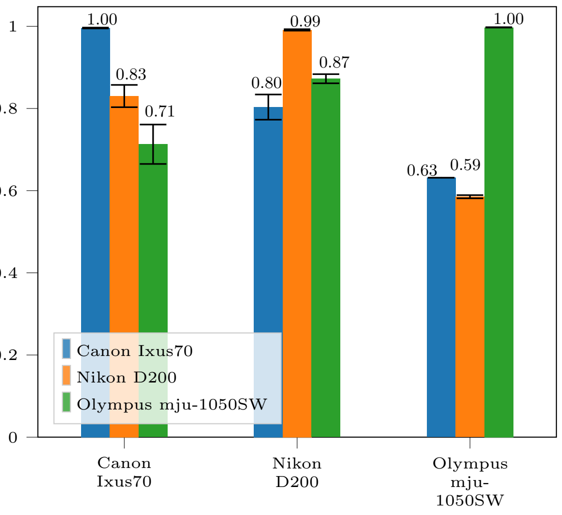

Figure 12: Mean prediction confidence to images from different devices of same model.

vices. We can say that our CNN-based camera model classifier is robust to the device-dependent characteristics.

4 Conclusion

In this report, we presented how a CNN-based camera model classifier behaves in the presence of three different types of data, namely, unseen data, post-processed data, and data from different devices.

By suppressing image content and extracting high frequency traces from camera using constrained convolutional layer, the CNN architecture has at least 99.69% accuracy as well as mean confidence of 1 for the correct label in the baseline experiment. It makes our results independent to image's scenes and objects. However, we showed that our CNN is not robust to the out-of-distribution data in most of the experiments. It tends to classify unseen model images to the model with the same manufacturer as the input and makes decision with high confidence regardless of the ground truth. In the post-processing experiment, we have demonstrated that our CNN is robust to JPEG compression artifacts only for Nikon D200, but it doesn't generalize to the other two models. Furthermore, the CNN tends to classify noisy images as they are from Canon Ixus70 and blurred images as they are from Nikon D200 with high confidence. The only exception is the device-dependent characteristics experiment, our classifier shows similar accuracy and mean confidence compared to the baseline experiment, which means it is robust to device-dependent characteristics.

These are preliminary experiments. There are many aspects to be further investigated in the future work, such as the method of evaluating the prediction confidence, effects of more different image manipulation techniques to the robustness and effects of camera parameters while shooting.

References

[1] Belhassen Bayar and Matthew C. Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, page 5–10, New York, NY, USA, 2016. Association for Computing Machinery.

[2] Belhassen Bayar and Matthew C Stamm. Design principles of convolutional neural networks for multimedia forensics. *Electronic Imaging*, 2017(7):77–86, 2017.

[3] Thomas Gloe and Rainer Böhme. The "dresden image database" for benchmarking digital image forensics. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, page 1584–1590, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605586397. doi: 10.1145/1774088.1774427. URL `https://doi.org/10.1145/1774088.1774427`.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 448–456. JMLR. org, 2015.

[5] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.