Double JPEG Detection in Mixed JPEG Quality Factors using Deep Convolutional Neural Network

Jinseok Park[1], Donghyeon Cho[2], Wonhyuk Ahn[1], Heung-Kyu Lee[1]

[1]School of Computing, [2]Electrical Engineering,
Korea Advanced Institute of Science and Technology (KAIST)
jspark@mmc.kaist.ac.kr, cdh12242@gmail.com, whahnize@kaist.ac.kr,
heunglee@kaist.ac.kr(corresponding author)

Abstract. Double JPEG detection is essential for detecting various image manipulations. This paper proposes a novel deep convolutional neural network for double JPEG detection using statistical histogram features from each block with a vectorized quantization table. In contrast to previous methods, the proposed approach handles mixed JPEG quality factors and is suitable for real-world situations. We collected real-world JPEG images from the image forensic service and generated a new double JPEG dataset with 1120 quantization tables to train the network. The proposed approach was verified experimentally to produce a state-of-the-art performance, successfully detecting various image manipulations.

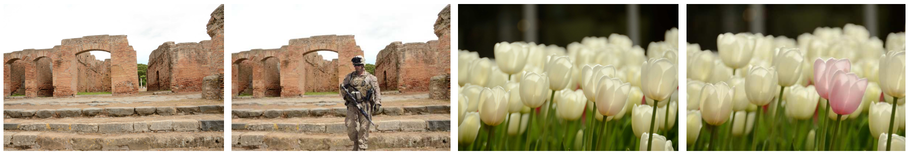
Keywords: Image forensics, Image manipulation, Fake images, Double JPEG, Convolutional Neural Networks

1 Introduction

With the development of digital cameras and relevant technology, digital images can be captured from anywhere, posted online, or sent directly to friends through various social network services. People tend to think that all such immediately posted digital images are real, but many images are fake, having been generated by image-editing programs such as Photoshop.

Image manipulation is easy but can have significant impact. The left two images of Fig. 1 show normal and spliced images implying an unrelated person may have been present somewhere. It is difficult to determine whether the spliced image is real or not by the naked eye. Such artificially created images spread distorted information and can cause various societal effects. Politicians and entertainers, for example, are particularly vulnerable to image manipulation, where persons can use composite images to undermine their reputation. The right two images of Fig. 1 show another manipulation example. The specific region of the image was replaced by different colors, which gives a different impression to the original image.

Thus, image manipulations can be applied to any image, and it is not easy to authenticate the image visually. Researchers have been developing image forensic



(a) Normal (b) Manipulated (c) Normal (d) Manipulated

Fig. 1. Two examples of image manipulations. The left two images show normal and spliced images. A soldier was extracted from another image and pasted into the normal image. The right two images show normal and color-modified images. The color of some tulips has been changed.

techniques for many years to distinguish fake images to overcome these problems and restore digital image credibility [1, 2].

Image forensic technology is categorized into two types. The first type target a specific manipulation and detects it. Many studies have developed detection methods for various manipulations, such as splicing [3–7], copy-move [8–11], color modification [12, 13], and face morphing [14]. Detection techniques based on these operation types are well suited to specific situations, where the target manipulation(s) has been applied. However, they cannot be applied generally because there are many image transformations aside from those considered, and images are often subject to multiple manipulations, where the order of operation is also significant.

The second approach detects remaining traces that occur when capturing the image by digital camera. In the digital image process, light passes through the camera lens and multiple filters and impacts on a capture array to produce pixel values that are stored electronically. Thus, the images include traces with common characteristics. Several image manipulation detection methods have been proposed to detect such traces [15–17], including detecting the interpolation operation generated by the color filter array [18, 19] and resampling traces generated during image manipulation [20–22].

Image forensic techniques using image acquisition traces have the advantage that they can be commonly applied to various image manipulations. However, the approach is almost impossible to use in a real image distribution environment. Although various traces are evident in uncompressed images, they are all high-frequency signals. Most digital images are JPEG compressed immediately when taken or compressed when they are uploaded online, which eliminates or modifies high frequency signals within the image.

Although JPEG compression removes many subtle traces, quantization, an essential part of the JPEG compression process, also leaves traces, and methods have been proposed to use these traces to detect image manipulations. Since JPEG is a lossy compression, image data differs between single and double image compression because the quantization tables are not unique (e.g., they strongly depend on the compression quality setting) [23].

Lukas et al. showed that the first compression quantization table could be estimated to some extent from a doubly compressed JPEG image and used to detect image manipulation [24]. Various double JPEG detection methods have been subsequently proposed. However, existing double JPEG detection methods only consider specific situations rather than a general solution. Therefore, this paper proposes detecting double JPEG compression for general cases with mixed quality factors to detect image manipulations.

The contributions can be summarized as follows. 1) We created a new double JPEG dataset suitable for real situations based on JPEG images obtained from two years of an image forensic service. 2) We propose a novel deep convolutional neural network (CNN, or ConvNet) structure that distinguishes between single and double JPEG blocks with high accuracy under mixed quality factor conditions. 3) We show that the proposed system can detect various image manipulations under a situation similar to one in the real-world.

2 Related Work

This section introduces current double JPEG methods and describes their limitations.

Early double JPEG detection: Early double JPEG detection methods extracted hand-crafted features from discrete cosine transformation (DCT) coefficients to distinguish between single and double JPEG images. Fu et al. found that Benford's rule occurs for JPEG coefficients and suggested it could be used to verify image integrity [25]. Li et al. proposed a method to detect double JPEG images by analyzing the first number of DCT coefficients [26].

In contrast to previous methods that assessed a double JPEG using the entire image, Lin et al. proposed a method to detect image manipulations from the DCT coefficients for each block [27], and Farid et al. proposed a method to detect partial image manipulations through JPEG ghost extraction [28]. These methods exploited the fact that manipulated JPEG images have different characteristics for each block.

Fig. 2 shows how some blocks are single or double JPEG across a manipulated image. JPEG compression is quantized in 8×8 block units. If the first and the second quantization tables are different, the distribution of the corresponding DCT coefficients differs from the distribution of the DCT coefficients of the JPEG compressed once. When the image is saved to JPEG format after changing the value of a specific region of a JPEG image, the distribution of the DCT coefficients in the region becomes similar to the DCT coefficient of the single JPEG. This is because when the pixels of the region are changed, the quantization interval of the DCT coefficient that already exists disappears.

Bianchi et al. investigated various double JPEG block detection aspects, and proposed an image manipulation detection method based on analyzing DCT coefficients [29]. They also discovered that double JPEG effects could be classified into two cases, aligned and non-aligned [30]. Chen et al. showed that periodic

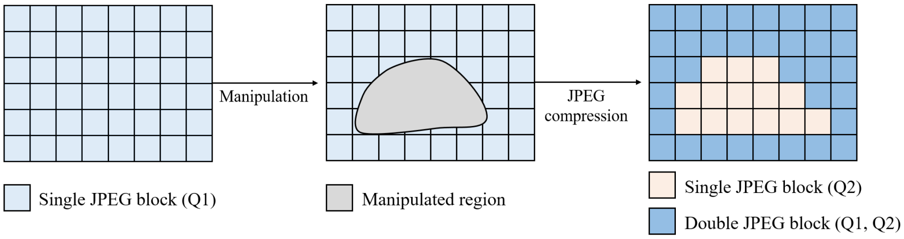


Fig. 2. The detection of image manipulations using double JPEG detection algorithm. The normal image ordinarily has single JPEG characteristics of quantization table 1 (Q1). If the normal image is manipulated and re-compressed with quantization table 2 (Q2), the manipulated region has the single JPEG characteristics of Q2. In contrast, the other part has double compression characteristics. We can find the suspicious area by separating single JPEG blocks and double JPEG blocks.

patterns appear in double JPEG spatial and frequency domains and proposed an image manipulation detection method based on this effect [31].

Double JPEG detection using ConvNets: Two neural network based methods have been recently proposed to improve current hand-crafted feature based double JPEG detection performance.

Wang et al. showed double JPEG blocks can be detected using ConvNets. They experimentally demonstrated that CNNs could distinguish single and double JPEG blocks with high accuracy when histogram features were inserted into the network after extracting from the DCT coefficients [32]. Subsequently, Barni et al. found that ConvNets could detect double JPEG block with high accuracy when the CNNs took noise signal or histogram features as input [33].

Limitations of current double JPEG detecting methods: Although double JPEG detection performance has greatly improved, current detection methods have major drawbacks for application in real image manipulation environments. Current methods can only perform double JPEG detection for specific JPEG quality factor states such as in the case where the first JPEG quality (Q1) is 90 and the second JPEG quality (Q2) is 80. However, actual distributed JPEG images can have very different characteristics with a very diverse mixture of JPEG quality parameters. Images are JPEG compressed using not only the standard quality factor (SQ) but also each individual program's JPEG quality factor.

3 Real-World Manipulated Images

We have operated a public forensic website for two years to provide a tool for determining image authenticity. Thus, we could characterize real-world manipulated images. This section introduces the characteristics of requested images and the method employed to generate the new dataset used to develop the generalized double JPEG detection algorithms.

Table 1. Summarization of requested images through the forensic website over two years. 77.95% images of JPEG format, and 41.77% images with the nonstandard quantization table. Q represents quality factor. Each Q corresponds to a different quantization table.

The number of images	127,874			
Manipulation type and region	Unknown			
File type	JPG(+JPEG)	PNG	BMP	TIF(+TIFF)
Percent (%)	77.95	20.67	1.21	0.18
JPEG quality type	Standard Q (0-100)		Nonstandard Q	
Percent (%)	58.22		41.77	
The number of Q types	1170			

3.1 Requested images

Table 1 shows a total of 127,874 images were requested to inspect authenticity over two years. As a result of analyzing the requested image data, the JPEG format was found to be the most requested (77.95%), followed by PNG (20.67%).

JPEG images: As discussed above, JPEG compression quantizes DCT coefficients using a predefined 8×8 JPEG quantization table. Previous studies have assumed that all JPEG images are compressed with standard quality factors, but even Photoshop, the most popular image-editing program, does not use the standard quality factor. Rather, Photoshop uses 12-step quantization tables that do not include the standard quality factor. Among the 99,677 JPEG images from the forensics website, only 58.22% had standard quality factors from 0 to 100, with 41.78% using nonstandard quantization tables. In total, 1170 quantization tables were identified, including 101 different standard quantization tables.

3.2 Generating new datasets

We generated single and double JPEG blocks of 256×256 in size using collected quantization tables extracted from 99,677 collected JPEG images[1]. Since images with standard quality factors of less than 50 degraded severely, we only considered standard quality factors from 51 to 100; that is, we created a compressed image using a total of 1120 quantization tables.

Since it was not known in what state the collected JPEG images were uploaded, they could not be directly used to generate datasets. For this reason, we used 18,946 RAW images from 15 different camera models in the three raw image datasets [34–36] and split the images into a total of 570,215 blocks. The single JPEG blocks were produced by compressing each RAW block with a randomly chosen quantization table, and the double JPEG blocks were produced by further compression with another random quantization table.

Comparison with existing double JPEG datasets: Current double JPEG detection methods were developed from data generated from a very limited range

[1] https://sites.google.com/view/jspark/home/djpeg

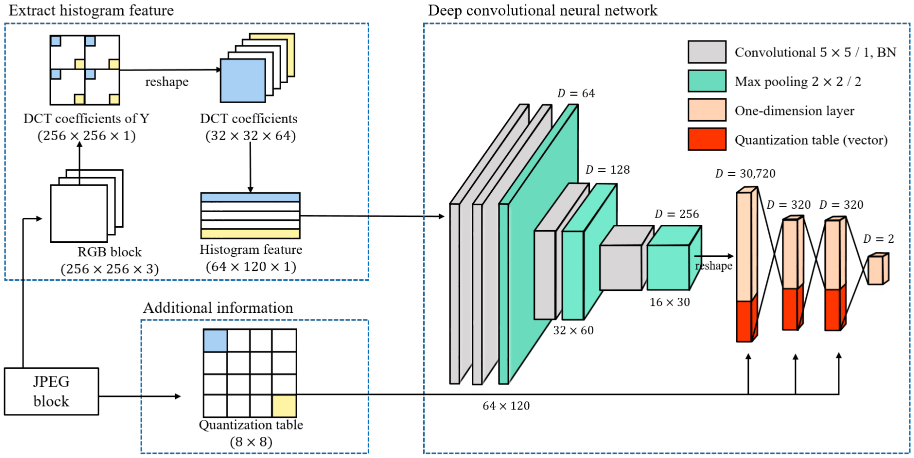


Fig. 3. A network architecture to distinguish between single and double JPEG blocks. The block is transformed in the DCT domain of the Y channel, and its histogram feature is forwarded to the network. The quantization table from the JPEG header is concatenated with the fully connected layer.

of JPEG quality factors, from 50 to 100, with predefined first quality factors, rather than mixed quality factors. In contrast, the double JPEG dataset we created differs from previous datasets as follows.

- We collected 1120 different quantization tables from actual requested images.
- The images were compressed using 1120 quantization tables.
- Data was generated by mixing all quality factors.

4 Double JPEG Block Detection

This section introduces the new double JPEG block detection method using a CNN and describes the detection of manipulated regions within an image.

4.1 Architecture

The proposed CNN takes histogram features and quantization tables as inputs. We first explain how to construct the input data and then provide the CNN details.

Histogram features: Since JPEG compression changes the statistical properties of each block rather than the semantic information of the entire image, DCT coefficient statistical characteristics were employed rather than the RGB image as CNN input [33].

Fig. 3 shows how the RGB blocks were converted into histogram features. RGB blocks were converted into YCbCr color space and DCT coefficients of the Y channel calculated for each 8×8 block. Thus, the DCT coefficients had the same size as the RGB block and frequency information was saved for every position skipped by 8 in the horizontal and vertical directions. This is the same as JPEG compression. We then collected data D with the same frequency component for each channel. The total number of channels was 64 (one DC and 63 AC channels), where each channel is represented by D_c. The process to calculate D from Y can be accomplished in a single convolutional (stride is 8) operation as below:

$$D = conv_8(Y, B), \tag{1}$$

where B is a $8 \times 8 \times 64$ matrix set of 8×8 DCT basis functions. D has a 1/8 width and height (N_W and N_H, respectively) compared to the input block and 64 channels. Thus, the size of D is $32 \times 32 \times 64$.

After calculating D, we extracted histogram features from each channel. The chosen histogram feature was the percentage of values in each channel relative to the total amount of data, where we set the histogram range as $b = [-60, 60]$, which was determined experimentally to provide the best performance. To extract histogram features, we first subtracted b from D_c and applied the sigmoid function after multiplying by γ, which provided a sufficiently large positive value if each $D_c - b$ was positive and a sufficiently large negative value if each $D_c - b$ was negative. Thus, we set $\gamma = 10^6$. Therefore,

$$S_{c,b} = sigmoid(\gamma * (D_c - b)), \tag{2}$$

where $S_{c,b}$ has the same width and height as D_c, and each value of $S_{c,b}$ is close to zero or one.

We then calculated $a_{c,b}$ by averaging $S_{c,b}$ and generated H features for all b and c,

$$a_{c,b} = \frac{1}{N_W * N_H} \sum_{i=1}^{N_H} \sum_{j=1}^{N_W} S_{c,b}(i,j), \tag{3}$$

and

$$H = \left\{ h | h_{c,b} = a_{c,b+1} - a_{c,b}, \quad \forall c, b \right\}, \tag{4}$$

where H is a two-dimensional $|c| \times |b|$ matrix and each raw of H is a histogram of channel c of the DCT coefficients. This operation was not part of learning, because there were no weights, but was implemented as a network operation for end-to-end learning.

Quantization table: The JPEG image file's header contain the quantization table in the form of an 8×8 matrix, which is used for the quantization and dequantization of DCT coefficients. Quantization table information is not required for conventional double JPEG detection, since the JPEG quality factor is

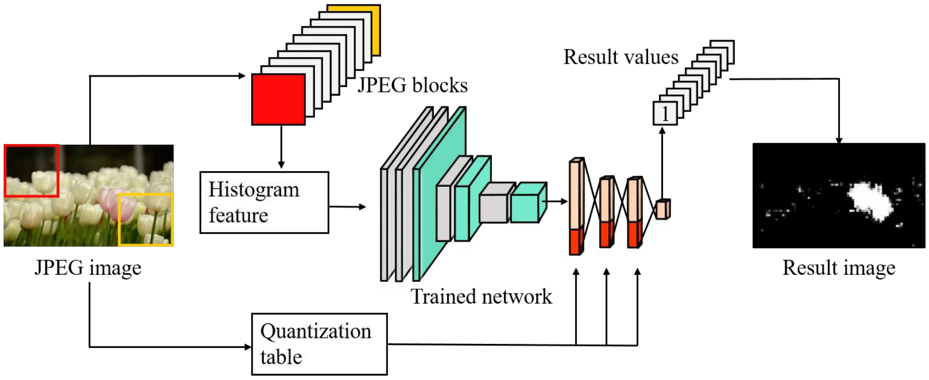


Fig. 4. The process of detecting the manipulated regions in a JPEG image. Through the sliding window, the manipulated regions are identified by detecting single and double JPEG blocks.

usually fixed. However, this paper considers mixed JPEG quality factors; thus, the quantization table will facilitate single and double JPEG assessment. For a double JPEG image, only the second quantization table is stored in the file.

To input the quantization table into the network, we reshaped it into a vector and then merged the vector with the activations of the last max pooling layer and two fully connected layers as shown in Fig. 3 (right block). The ability of the network to distinguish between single and double JPEG blocks was dramatically improved by including quantization table information.

Deep ConvNet: The deep ConvNet received the histogram features and quantization table inputs and assessed if the corresponding data was single or double JPEG compressed. The network consisted of four convolutional layers, three max pooling layers, and three fully connected layers, as shown in Fig. 3 (right block). The quantization table vector was combined with the last max pooling layer and two fully connected layer activations. The final network output was a 2×1 vector, y, where $y = [1; 0]$ for a single block and $y = [0; 1]$ for a double block. The loss, L, was calculated from cross entropy,

$$L = -(1-p) * log(\frac{e^{y_0}}{e^{y_0} + e^{y_1}}) - p * log(\frac{e^{y_1}}{e^{y_0} + e^{y_1}}), \quad (5)$$

where $p = 0$ if the input data is a single JPEG and $p = 1$ for a double JPEG.

4.2 Manipulated Region Detection

As mentioned in Fig. 2, when a specific part of a JPEG image was manipulated and then stored as JPEG again, the specific region had a single JPEG block property and the other region had a double JPEG block property.

Using this principle, to find the manipulated area, we extracted blocks from the whole image using a sliding window and determined if the block was single or double compressed using the trained deep ConvNet, as shown in Fig. 4. The sliding window's stride size had to be a multiple of 8 because the compression process was conducted in 8×8 block units. Thus, the compression traces aligned with the 8×8 blocks, and if we extracted blocks randomly they would have different properties.

Let $y(i,j)$ be the network output of the input block of location (i,j), then

$$R = \left\{ r | r_{i,j} = \frac{e^{y_0(i,j)}}{e^{y_0(i,j)} + e^{y_1(i,j)}}, \quad \forall i,j \right\}, \tag{6}$$

where, r is the probability the block was compressed once. R could be visualized, and where some regions appeared single compressed, and others appeared double compressed, only the single-compressed portion had been manipulated.

5 Experiments

This section compares the classification accuracy to detect double JPEG blocks using several state-of-the-art methods and compares the results of detecting manipulated images.

5.1 Comparison with the state-of-the-art

We divided double JPEG block detection into three parts: first, double JPEG detection using VGGNet [37], which has shown good performance in many computer vision applications; second, two networks specialized for double JPEG detection by Wang [32], and Barni [33]; third, detection results for the proposed network.

The experiments were performed using the dataset generated in Section 3, comprising 1,026,387 blocks for training and 114,043 blocks for testing. All experiments were conducted using TensorFlow 1.5.0 and GeForce GTX 1080, with an initial learning rate of 0.001 and an Adam optimizer.

VGG-16Net: Table 2, part 1 shows VGG-16Net detection performance directly using RGB blocks to distinguish between single and double JPEG blocks. VGG-16Net has previously shown good performance for object category classification, but could not distinguish between single compressed and double compressed JPEG blocks. This is because it is necessary to distinguish the statistical characteristics of DCT coefficients to detect double JPEG, but VGG-16Net uses the semantic information rather than the statistical characteristics of DCT coefficients.

Networks using histogram features: Two methods have been proposed with CNNs and histogram features to distinguish double JPEG. Wang et al. proposed histogram features for DCT values $[-5, 5]$ from nine DCT channels, comprising two one-dimensional convolutional layers, two max pooling layers, and three fully connected layers. Barni et al. also used histogram features but the network calculated histogram features within the ConvNets, collecting DCT values $[-50, 50]$

Table 2. Performance comparison between double JPEG detection ConvNets. All variants of proposed methods outperformed previous networks. ACC, TPR, and TNR represent the accuracy, true positive rate, and true negative rate, respectively, and positive means classifying a block as double JPEG. The network with the highest accuracy for each part is highlighted in red.

Method	Type	ACC(%)	TPR(%)	TNR(%)
VGG-16 [37]	RGB pixels	50.00	00.00	100.00
Wang [32]	Range - $[-5, 5]$	73.05	67.74	78.37
Barni [33]	Range - $[-50, 50]$	83.47	77.47	89.43
	Range - $[-60, 60]$	84.46	78.35	90.53
	Range - $[-70, 70]$	82.87	75.80	89.90
	Range - $[-80, 80]$	83.90	76.41	91.39
	Range - $[-90, 90]$	83.44	75.65	91.19
	Range - $[-100, 100]$	82.30	79.64	84.92
Proposed (+ Q table)	Last conv output	89.89	86.95	92.81
	First FC ouput	89.95	83.73	96.15
	Second FC output	86.79	82.90	90.67
	All three outputs	90.37	84.84	95.88
Proposed (# of conv layers)	4 conv layers	91.83	87.16	96.50
	5 conv layers	91.33	90.19	92.43
	6 conv layers	91.72	85.93	96.10
	7 conv layers	91.02	85.93	96.10
Proposed*	Optimization	92.76	90.90	94.59

from 64 DCT channels, and comprising three convolutional layers, three max pooling layers, and three fully connected layers.

Table 2, parts 2 and 3 shows that the Wang and Barni network classified single or double JPEG blocks with 73.05% and 83.47% accuracy, respectively. The Barni method extracts histograms over a wider range; thus, it has over 10% better performance due to the larger number of network layers. Compared with the VGG-16Net results, it is critical to use histograms with statistical features for double JPEG detection.

Additional experiments were conducted with the Barni network to investigate how accuracy varied with the histogram range. We tried to increase the histogram range to $[-100, 100]$, but we found that the accuracy was lower if the range was over $[-60, 60]$. Based on this phenomenon, it was estimated that most DCT coefficients were less than 60.

Proposed networks: The most important point of the proposed network is to include quantization table information in the neural network. We constrained the network structure to match the Barni network that has a $[-60, 60]$ histogram range and inserted quantization table information at three different locations to determine the optimal insertion point: each output of the final convolutional layer, of the first fully connected layer, and of second fully connected layer, as shown in Table 2, part 4. Even though only the quantization table was inserted, the accuracy was 5.43%, 5.52%, and 2.33% higher than the Barni network that had a $[-60, 60]$ histogram range network according to the insertion points. We also inserted the quantization table into all three locations, as shown in the final row of Table 2, part 4, producing the best accuracy (90.37%).

Table 2, part 5 compares the proposed network performance according to convolutional layer depth. Since the previous network used three convolutional layers, we increased the depth from four to seven layers. Increasing the number of convolutional layers to four provided a significant increase in accuracy (1.46% improvement), but there was no subsequent significant improvement for five or more layers because the histogram feature already compressed the statistical data characteristics sufficiently.

The final optimal network had four 5×5 convolutional layers , three max pooling layers and three fully connected layers as shown in Fig. 3. The quantization table information was combined with the output of the last max pooling layer, the output of the first fully connected layer, and the output of the second fully connected layer. All convolutional layers were used with batch normalization [38]. The optimization network reached 92.76% accuracy, as shown in Table 2, part 6.

5.2 Manipulated Region Detection

This section shows the results of image manipulation detection using the proposed network. The 14 images used in the experiments were manipulated in the following order. First, we generated single JPEG images using 1120 different randomly selected quantization tables. Second, we manipulated images by splicing, copy-move, color changing, brightness changing, interpolation, blurring, and resizing using Photoshop. Third, we saved the manipulated images using different randomly selected quantization tables apart from the first one. All manipulated region detection experiments were performed in 32 strides.

Results for copy-move and splicing manipulations: Fig. 5 shows the six results for the copy-move and splicing manipulations. The top two lines show the copy-move manipulations and detection results. Two manipulated images were made by copying the windows and cherry blossoms in the image and then pasting them to another location in the same image. Because copy-move operations are performed within the same image, natural manipulation is possible. The proposed network found single JPEG blocks near the ground truth; however, the Barni network incorrectly detected many double JPEG blocks as single JPEG blocks.

The bottom four lines show the splicing manipulations and detection results. Splicing is one of the most important detection operations because it can completely change the meaning of an image. We pasted four people into four images that were not related to them and applied the blur filter to object edges. The proposed network properly detected four manipulated regions, but, the Barni network detected only one region.

Results for local manipulations: Fig. 6 shows six results for local manipulations. The top three line of manipulated images were made by color trans-

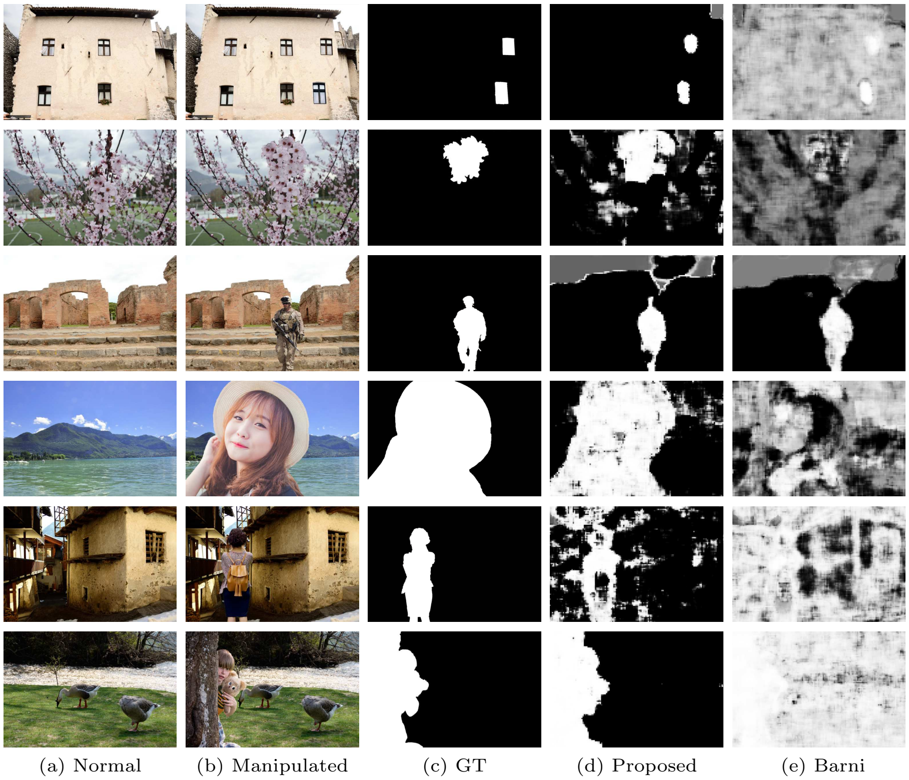


Fig. 5. Examples of copy-move and image splicing detection. (a) and (b) are normal and manipulated images, (c) is the ground truth, (d) is the results of the proposed network, and (e) is the results of the Barni network. The top two rows of images show the copy-move manipulation. The bottom fourth-row images show the image splicing manipulation.

formation and changing the brightness. Each image became other images with completely different information by changing the color of the tulips, houses, and cars. In the case of the tulip image, the proposed network correctly found a single JPEG area, whereas the Barni network determined that all areas were a single JPEG. The proposed network showed better performance for the second and third manipulated images.

We erased the banner photos in the building using a content-aware interpolation method, blurred a model's face, and resized the boat. The Barni network distinguished some of the manipulated regions, but there were many false negatives. On the other hand, the proposed network detected single JPEG regions with much high accuracy.

F-measure: To numerically compare the manipulation region detection capabilities, we conducted quantitative experiments on two manipulations—copy-move

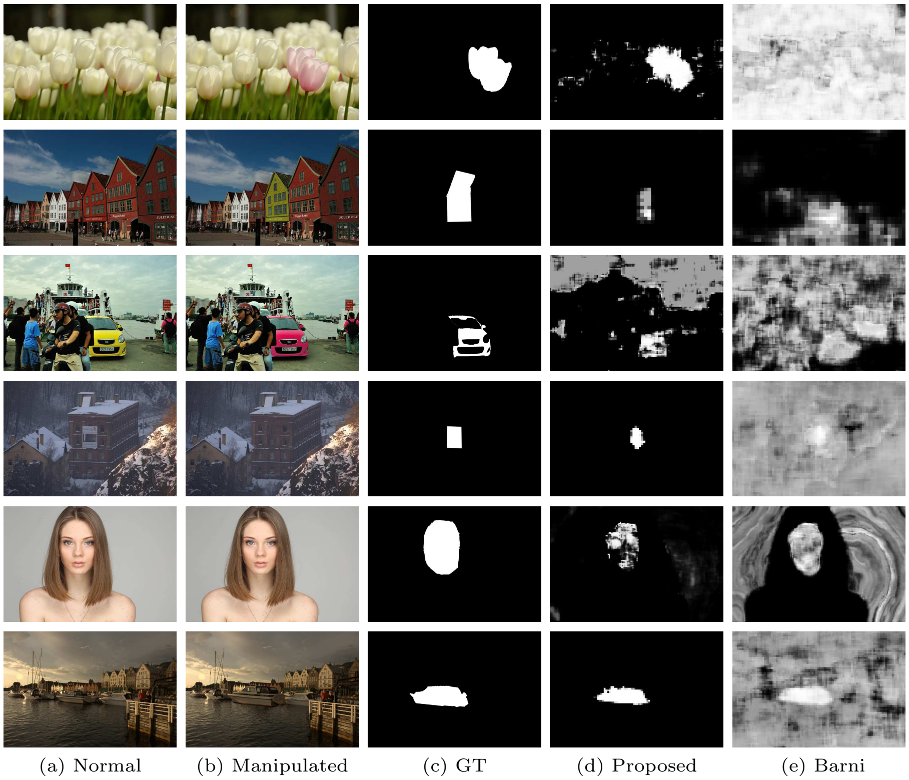


Fig. 6. Examples of local manipulation detection. (a) and (b) are normal and manipulated images, (c) is the ground truth, (d) is the results of the proposed network, and (e) is the results of the Barni network. The top third-row images underwent a change in color, and the bottom third-row images were changed by other local manipulations.

and blurring. We generated 2100 images of 1024 × 1024 in size for each manipulation with raw image datasets [34–36]. In the case of the copy-move manipulation, a patch of 544 × 544 in a random position was copied and pasted into a random location in the same image. In the case of the blur manipulation, a blur filter ($\sigma = 2$) was applied to a 544 × 544 area of a random position in the image. JPEG compression was performed using 1120 quantization tables in the same manner as 14 representative manipulation images. Table 3 shows the detection results (F-measure) for the copy-move and blur manipulations. That of the proposed network was approximately 0.12 higher than that of the Barni network.

Failure results and analysis: In some cases, manipulation regions were not properly detected. The second line of Fig. 5 shows that both the proposed and the Barni networks had false negatives because the pixel values in the sky were saturated and only low frequencies were present. In addition, if single JPEG quality and double JPEG quality were the same or there was little difference,

Table 3. F-measures for two manipulations using the proposed network and the Barni network, respectively.

	Proposed	Barni [33]
Copy-move (F-measure)	0.7595	0.6323
Blurring (F-measure)	0.7783	0.6450

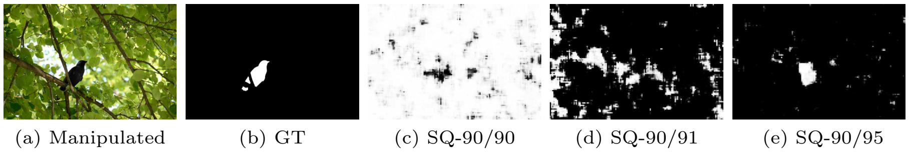

(a) Manipulated (b) GT (c) SQ-90/90 (d) SQ-90/91 (e) SQ-90/95

Fig. 7. Detecting results according to changing the second quality factor. (a) is a manipulated image by changing the color of a bird, (b) is the ground truth and, (c) - (e) are the detection results for images generated according to each quality factor.

it was impossible to detect the operation area. Because the DCT coefficients of a single JPEG block and a double JPEG block were almost identical, the network could not distinguish between the two classes. Fig. 7 shows the detection results according to changing the second quality factor (standard quality factor). Although it was impossible to detect image manipulation with the same quality factor, as the quality factor difference increased, the proposed network could detect the manipulation region.

6 Conclusion

Current double JPEG detection methods only work in very limited situations and cannot be applied to real situations. To overcome this limits, we have created a new dataset using JPEG quantization tables from actual forensic images and designed a novel deep CNN for double JPEG detection using statistical histogram features from each block with a vectorized quantization table. We have also proven that the proposed network can detect various manipulations with mixed JPEG quality factors.

Acknowledgements

This work was supported by the Institute for Information & communications Technology Promotion (IITP) grant funded by the Korean government (MSIP) (2017-0-01671, Development of high reliability image and video authentication service for smart media environment).

References

1. Piva, A.: An overview on image forensics. ISRN Signal Processing **2013** (2013)

2. Stamm, M.C., Wu, M., Liu, K.R.: Information forensics: An overview of the first decade. IEEE Access **1** (2013) 167–200
3. Shi, Y.Q., Chen, C., Chen, W.: A natural image model approach to splicing detection. In: Proceedings of the 9th workshop on Multimedia & security, ACM (2007) 51–62
4. Chen, W., Shi, Y.Q., Su, W.: Image splicing detection using 2-d phase congruency and statistical moments of characteristic function. In: Security, Steganography, and Watermarking of Multimedia Contents IX. Volume 6505., International Society for Optics and Photonics (2007) 65050R
5. Cozzolino, D., Poggi, G., Verdoliva, L.: Splicebuster: A new blind image splicing detector. In: Information Forensics and Security (WIFS), 2015 IEEE International Workshop on, IEEE (2015) 1–6
6. Rao, Y., Ni, J.: A deep learning approach to detection of splicing and copy-move forgeries in images. In: Information Forensics and Security (WIFS), 2016 IEEE International Workshop on, IEEE (2016) 1–6
7. Chen, C., McCloskey, S., Yu, J.: Image splicing detection via camera response function analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017) 5087–5096
8. Fridrich, A.J., Soukal, B.D., Lukáš, A.J.: Detection of copy-move forgery in digital images. In: in Proceedings of Digital Forensic Research Workshop, Citeseer (2003)
9. Ryu, S.J., Lee, M.J., Lee, H.K.: Detection of copy-rotate-move forgery using zernike moments. In: International Workshop on Information Hiding, Springer (2010) 51–65
10. Li, J., Li, X., Yang, B., Sun, X.: Segmentation-based image copy-move forgery detection scheme. IEEE Transactions on Information Forensics and Security **10**(3) (2015) 507–518
11. Zhou, Z., Wang, Y., Wu, Q.J., Yang, C.N., Sun, X.: Effective and efficient global context verification for image copy detection. IEEE Transactions on Information Forensics and Security **12**(1) (2017) 48–63
12. Choi, C.H., Lee, H.Y., Lee, H.K.: Estimation of color modification in digital images by cfa pattern change. Forensic science international **226**(1-3) (2013) 94–105
13. Hou, J.U., Lee, H.K.: Detection of hue modification using photo response nonuniformity. IEEE Transactions on Circuits and Systems for Video Technology **27**(8) (2017) 1826–1832
14. Zhou, P., Han, X., Morariu, V.I., Davis, L.S.: Two-stream neural networks for tampered face detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE. (2017) 1831–1839
15. Swaminathan, A., Wu, M., Liu, K.R.: Digital image forensics via intrinsic fingerprints. IEEE transactions on information forensics and security **3**(1) (2008) 101–117
16. Cao, H., Kot, A.C.: Accurate detection of demosaicing regularity for digital image forensics. IEEE Transactions on Information Forensics and Security **4**(4) (2009) 899–910
17. Stamm, M.C., Liu, K.R.: Forensic detection of image manipulation using statistical intrinsic fingerprints. IEEE Transactions on Information Forensics and Security **5**(3) (2010) 492–506
18. Popescu, A.C., Farid, H.: Exposing digital forgeries in color filter array interpolated images. IEEE Transactions on Signal Processing **53**(10) (2005) 3948–3959
19. Ferrara, P., Bianchi, T., De Rosa, A., Piva, A.: Image forgery localization via fine-grained analysis of cfa artifacts. IEEE Transactions on Information Forensics and Security **7**(5) (2012) 1566–1577

20. Popescu, A.C., Farid, H.: Exposing digital forgeries by detecting traces of resampling. IEEE Transactions on signal processing **53**(2) (2005) 758–767
21. Kirchner, M., Gloe, T.: On resampling detection in re-compressed images. In: Information Forensics and Security, 2009. WIFS 2009. First IEEE International Workshop on, IEEE (2009) 21–25
22. Mahdian, B., Saic, S.: Blind authentication using periodic properties of interpolation. IEEE Transactions on Information Forensics and Security **3**(3) (2008) 529–538
23. Pennebaker, W.B., Mitchell, J.L.: JPEG: Still image data compression standard. Springer Science & Business Media (1992)
24. Lukáš, J., Fridrich, J.: Estimation of primary quantization matrix in double compressed jpeg images. In: Proc. Digital Forensic Research Workshop. (2003) 5–8
25. Fu, D., Shi, Y.Q., Su, W.: A generalized benford's law for jpeg coefficients and its applications in image forensics. In: Security, Steganography, and Watermarking of Multimedia Contents IX. Volume 6505., International Society for Optics and Photonics (2007) 65051L
26. Li, B., Shi, Y.Q., Huang, J.: Detecting doubly compressed jpeg images by using mode based first digit features. In: Multimedia Signal Processing, 2008 IEEE 10th Workshop on, IEEE (2008) 730–735
27. Lin, Z., He, J., Tang, X., Tang, C.K.: Fast, automatic and fine-grained tampered jpeg image detection via dct coefficient analysis. Pattern Recognition **42**(11) (2009) 2492–2501
28. Farid, H.: Exposing digital forgeries from jpeg ghosts. IEEE transactions on information forensics and security **4**(1) (2009) 154–160
29. Bianchi, T., Piva, A.: Analysis of non-aligned double jpeg artifacts for the localization of image forgeries. In: Information Forensics and Security (WIFS), 2011 IEEE International Workshop on, IEEE (2011) 1–6
30. Bianchi, T., De Rosa, A., Piva, A.: Improved dct coefficient analysis for forgery localization in jpeg images. In: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE (2011) 2444–2447
31. Chen, Y.L., Hsu, C.T.: Detecting recompression of jpeg images via periodicity analysis of compression artifacts for tampering detection. IEEE Transactions on Information Forensics and Security **6**(2) (2011) 396–406
32. Wang, Q., Zhang, R.: Double jpeg compression forensics based on a convolutional neural network. EURASIP Journal on Information Security **2016**(1) (2016) 23
33. Barni, M., Bondi, L., Bonettini, N., Bestagini, P., Costanzo, A., Maggini, M., Tondi, B., Tubaro, S.: Aligned and non-aligned double jpeg detection using convolutional neural networks. Journal of Visual Communication and Image Representation **49** (2017) 153–163
34. Dang-Nguyen, D.T., Pasquini, C., Conotter, V., Boato, G.: Raise: A raw images dataset for digital image forensics. In: Proceedings of the 6th ACM Multimedia Systems Conference, ACM (2015) 219–224
35. Gloe, T., Böhme, R.: The dresden image database for benchmarking digital image forensics. Journal of Digital Forensic Practice **3**(2-4) (2010) 150–159
36. Bas, P., Filler, T., Pevnỳ, T.: break our steganographic system: The ins and outs of organizing boss. In: International Workshop on Information Hiding, Springer (2011) 59–70
37. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

38. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. (2015) 448–456