EURASIP Journal on Information Security

RESEARCH Open Access

Boosting CNN-based primary quantization matrix estimation of double JPEG images via a classification-like architecture

Benedetta Tondi[1*], Andrea Costanzo[1], Dequ Huang[2] and Bin Li[2]

Abstract

Estimating the primary quantization matrix of double JPEG compressed images is a problem of relevant importance in image forensics since it allows to infer important information about the past history of an image. In addition, the inconsistencies of the primary quantization matrices across different image regions can be used to localize splicing in double JPEG tampered images. Traditional model-based approaches work under specific assumptions on the relationship between the first and second compression qualities and on the alignment of the JPEG grid. Recently, a deep learning-based estimator capable to work under a wide variety of conditions has been proposed that outperforms tailored existing methods in most of the cases. The method is based on a convolutional neural network (CNN) that is trained to solve the estimation as a standard regression problem. By exploiting the integer nature of the quantization coefficients, in this paper, we propose a deep learning technique that performs the estimation by resorting to a simil-classification architecture. The CNN is trained with a loss function that takes into account both the accuracy and the mean square error (MSE) of the estimation. Results confirm the superior performance of the proposed technique, compared to the state-of-the art methods based on statistical analysis and, in particular, deep learning regression. Moreover, the capability of the method to work under general operative conditions, regarding the alignment of the second compression grid with the one of first compression and the combinations of the JPEG qualities of former and second compression, is very relevant in practical applications, where these information are unknown a priori.

Keywords: Image forensics, Double JPEG compression, Quantization matrix estimation, Deep learning for forensics, Convolutional neural networks

1 Introduction

Detection of double JPEG (DJPEG) compression is one of the most widely studied problems in image forensics, see for instance [1–3]. The interest of researcher in this topic is motivated by the fact that double compression can reveal important information about the past history of an image. Important information can be obtained by estimating the quality of the first JPEG compression, and, moreover, by estimating the primary quantization matrix used for the first JPEG compression. Given an image with several copy-pasted regions, it is possible to identify the different origin of the tampering by recognizing that they have been compressed first with different JPEG qualities, and more in general that they are characterized by different primary quantization matrices of the compression (while there is not a standard definition of the JPEG quality, the concept of quantization matrix is a standard one [4]).

*Correspondence: benedettatondi@gmail.com
[1]Department of Information Engineering and Mathematics, University of Siena, via Roma 56, 53100 Siena, Italy
Full list of author information is available at the end of the article

Several methods have been proposed in the literature for the estimation of the primary quantization matrix. Many of them exploit statistical modeling of DCT coefficients [5–8]. A common feature of all these approaches is that they work under particular operative conditions and settings about the relationship of the JPEG qualities of former and second compression, and the alignment of the 8×8 grid of the first compression with the second one. For instance, the method in [7] works only when the two compressions are aligned and the second quantization step is lower than the first one, that is when the quality of the second JPEG is higher than the quality of the first one (hence, $QF_1 < QF_2$ for the standard quantization matrix case). Similarly, the method in [6] is designed for the aligned JPEG case and cannot estimate the first quantization step when this is a divisor of the second one. A more general approach for the estimation in the aligned scenario has been recently proposed in [9]. The algorithm proposed in [5] can work both in the aligned and non-aligned cases; however, the performance drops when $QF_1 > QF_2$. Eventually, the method in [8] works in the non-aligned case only. Another drawback of such model-based techniques and approaches that rely on hand-crafted features is that their performance tend to decrease significantly when they are applied to small patches, that prevents the application of these methods for the local estimation of the quantization matrix of first compression, useful for tampering localization.

A modern method for primary quantization matrix estimation based on convolutional neural networks (CNN) has been recently proposed in [10]. Such method can work under very general operative conditions and on small (64×64) patches. This approach has been shown to outperform previous approaches, both in terms of accuracy and mean square error (MSE) of the estimation. In particular, in [10], the CNN is trained to minimize the squared difference between the predicted values of the quantization coefficients and the true values, hence the MSE of the estimation is minimized. Some works in the deep learning literature, however, shows that CNNs are better to solve classification than regression problems. CNNs can in fact achieve remarkably accurate results when trained to predict categorical variables, drawn from discrete probability distributions of data [11, 12]. Whenever possible, switching to a classification problem or consider hybrid methods that combine classification with regression has been shown to yield better results [13]. In [13], for instance, soft values are estimated by using a quantized regression architecture that first obtains a quantized estimate (using the softmax followed by the cross-entropy loss), and then refines it through regression of the residual.

Given the above, in this paper, we focus on improving the performance of CNN-based estimation of the primary quantization matrix by turning the regression into a classification-like problem, with the design of a suitable CNN architecture. Our approach starts from the observation that the quantization coefficients can only take integer values. Therefore, we design a structure such that the estimation of a vector of integer values, namely all the coefficients of the quantization matrix, can be performed in a classification-like fashion. For the implementation of the network (internal layers) we consider the same CNN architecture already considered in [10], yielding good results, namely DenseNet [14]. Similarly to [10], the CNN-based estimator is designed to work under very general operative conditions, i.e., when the second compression grid is either aligned or not with the one of first compression, and for every combinations of qualities of former and second compression. The capability of the method to work under both aligned and non-aligned DJPEG, and for all possible combinations of JPEG qualities, is very relevant in practical applications, where those information are not known a priori, thus making the adoption of dedicated method very impractical. Like in [10], we focus on the case where the estimation is carried out on small patches, that represents the most challenging scenario. As commonly done by the approaches from the literature of primary quantization matrix estimation, we assume that the test image is double compressed and do not consider the single JPEG scenario (in this case, quite reasonably, our method returns a quantization matrix that corresponds to a very high compression quality, the estimated coefficients being close to those of the quantization matrix for the case $QF = 100$. Said differently, the network regards the single compression with QF as a double compression with $QF_1 = 100$ and $QF_2 = QF$).

The rest of this paper is organized as follows: Section 2 recaps the main concepts of double compression and introduces the notation. The proposed method is described in Section 3. Then, Section 4 details the experimental methodology and the results are reported and discussed in Section 5. We conclude the paper with some final remarks in Section 6.

2 Basic concepts and notation

We denote by Q the quantization matrix, that is, the 8×8 matrix with the quantization steps of the DCT coefficients considered for the compression. A double compression occurs when an image compressed with a given Q_1 is decompressed (decompression involves de-quantization and inverse DCT) and compressed again with a second quantization matrix Q_2. The elements of Q_1 can be conveniently arranged in a vector of dimensionality 64, zig-zag ordered [4]. We denote by $\mathbf{q}_1$ such 64-dim vector built from Q_1. As commonly done in the literature [5–7, 10], we focus on the first elements of $\mathbf{q}_1$ and restrict the estimation to those coefficients. We denote with $(\mathbf{q}_1)_{N_c} = [q_{1,1}, q_{1,2}, ..., q_{1,N_c}]$ the vector of the first N_c coefficients of

$\mathbf{q}_1$. The coefficients at the medium-high DCT frequencies are in fact more difficult to estimate accurately, due to the stronger quantization usually applied to them; however, since these coefficients are not very discriminative (as they tend to be similar for most quantization matrices), their estimation is less important.

When a JPEG image is compressed a second time, the second compression grid can be either aligned or non-aligned to the first compression grid. The case of a non-aligned DJPEG corresponds to the most frequent scenario in practice. A grid misalignment occurs locally when image splicing is performed, that is, when a region of a single JPEG image is copy-pasted into another image, since in this case the alignment between the compression grids is rarely preserved. On a global level, we have a non-aligned DJPEG when the image is cropped in between the former and second compression stage, or some processing is applied causing a de-synchronization.

The quality of the JPEG compression is often summarized by many compression softwares by means of the JPEG Quality Factor (QF), whose values range from 0 to 100 (QF values lower than 50 however are seldom used in practice nowadays since they corresponds to extremely low qualities). A QF value specifies a quantization matrix Q (standard quantization matrix). For convenience, in the rest of the paper, we refer to the JPEG Quality Factor (QF). Note that, in principle, the proposed estimator can be applied to estimate any quantization matrix of former compression, be it standard and non-standard. In the rest of the paper, we denote with QF_2 the second compression QF and with QF_1 the former.

Like in [10], in this paper, we consider color DJPEG images (with 3 channels) and focus on the estimation of the primary quantization matrix Q_1 of the luminance channel, that is, the Y channel of the YC_bC_r color space [4].

3 Proposed classification-like CNN estimator

The proposed method starts from the observation that the $q_{1,i}$'s values are discrete values. In [10], where a regression problem is addressed to estimate $(\mathbf{q}_1)_{N_c}$, the values obtained at the output of the CNN are finally quantized to get the estimated vector $(\hat{\mathbf{q}}_1)_{N_c}$ (specifically, rounding is performed on each element of the output vector independently, yielding $\hat{q}_{1,i}$, $i = 1, ..., N_c$). However, it has been shown that for the estimation of discrete quantities (following then a categorical distribution), it is often preferable to resort to softmax followed by the cross-entropy loss, which is good for backpropagation (see [13]). Therefore, we propose to switch the regression to a classification-like problem. To do so, we consider a custom output layers structure with a basic loss function, and also with a refined loss function, described in the following.

3.1 General structure

The architecture that we considered for the internal layers of the CNN, and in particular, the feature extraction part, is a dense structure, namely the DenseNet backbone architecture [14], and is described in Section 4.1. In the following, we describe the specific structure of the proposed output layer.

In the proposed structure, each to-be-estimated coefficient $q_{1,i}$ ($i = 1, 2, .., N_c$) of discrete value is encoded as a one-hot vector. The dimensionality of the encoded vector is determined by all the possible values that $q_{1,i}$ may take. Assuming that the quality of the image cannot be too low, in fact, for every i, the estimated coefficient $\hat{q}_{1,i}$ may take a limited number of values, that is, $1 \leq \hat{q}_{1,i} \leq q^M_{1,i}$. For simplicity, we set $q^M_{1,i}$ equal to the corresponding value of the i-th coefficient when $QF_1 = 50$ (minimum quality of the JPEG considered)[1]. To get the desired output, we set the logit level output to a size $\left[q^M_{1,1} + q^M_{1,2} \cdots + q^M_{1,N_c}\right]$; then, the softmax is applied block-based on each N_c block, where each block has $q^M_{1,i}$ inputs, $i = 1, 2, .., N_c$.

For training the network, we consider the following basic custom loss:

$$\mathcal{L}(\mathbf{x}) = \frac{1}{N_c} \sum_{i=1}^{N_c} \left(\sum_{j \in [1:q^M_{1,i}]} y_{ij} \log(f_{ij}(\mathbf{x})) \right), \qquad (1)$$

where $y_{i,j}$ denotes the ground-truth label corresponding to $q_{1,i}$. According to Eq. (1), a cross-entropy loss is first computed on each block separately, then the loss is defined as the sum of all the cross-entropy loss terms.

Figure 1 illustrates the scheme of the CNN considered for the estimation with specific focus on the output layer. In the figure, $\mathbf{y}$ denotes the ground-truth vector of the N_c one-hot encoded vectors, having the dimensionality $\left[q^M_{1,1} + q^M_{1,2} + \cdots + q^M_{1,N_c}\right]$, and $f(\mathbf{x})$ the output soft vector of the CNN, having the same dimensionality of $\mathbf{y}$. Formally, $\mathbf{y} = \mathbf{y}_1 \oplus \mathbf{y}_2 \oplus \cdots \oplus \mathbf{y}_{N_c}$, where $\oplus$ denotes the horizontal concatenation, that is,

$$\mathbf{y} = \left[y_{11}, y_{12}, ..., y_{1q^M_{1,1}}, y_{21}, \cdots y_{N_c1}, ..., y_{N_cq^M_{1,N_c}}\right], \qquad (2)$$

and, similarly, $f(\mathbf{x}) = [f_1(\mathbf{x}) \oplus \cdots f_{N_c}(\mathbf{x})]$ where $f_i(\mathbf{x}) = (f_{ij}(\mathbf{x}))_{j=1}^{q^M_{1i}}$.

As we said, $q^M_{1,i}$, for every i, is determined considering the value assumed by the i-th coefficient in the quantization matrix corresponding to the lowest QF_1 considered (that we set to 50 in our experiments). Then, the final estimated vector $(\hat{\mathbf{q}}_1)_{N_c}$ is given by

[1]This corresponds to assume that the former JPEG quality is always higher than or equal to $QF_1 = 50$, which is often the case in practice, thus not representing a big restriction (as a consequence, Q_1 matrices corresponding to lower qualities are not correctly estimated).

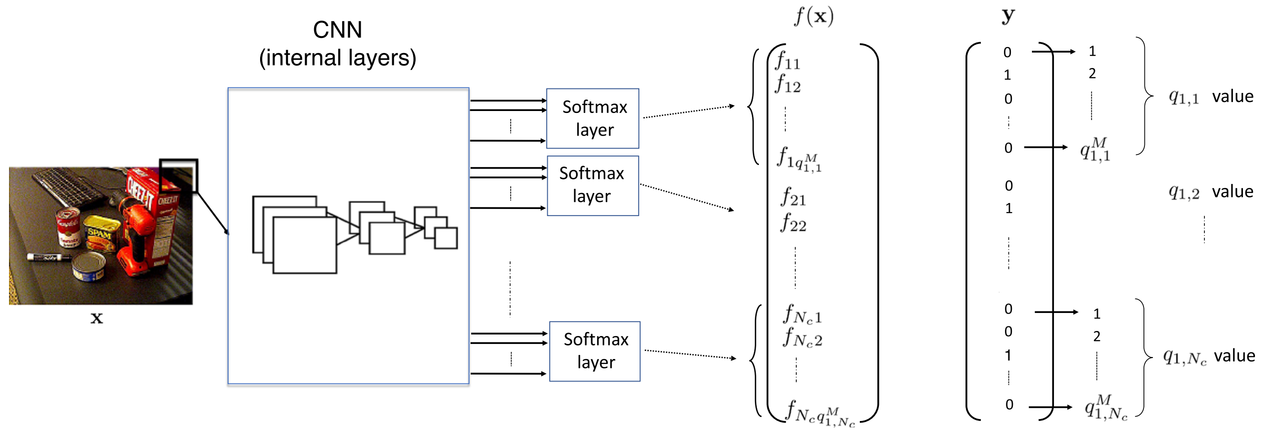

Fig. 1 Scheme of the CNN classification-like structure considered in this paper

$$\hat{q}_{1,i} = \arg\max_j f_{ij}(\mathbf{x}), \quad i = 1, ..., N_c. \tag{3}$$

3.2 Refined loss function

For a given image $\mathbf{x}$, and final predicted vector $(\hat{\mathbf{q}}_1)_{N_c}$, the accuracy of the estimation is averaged over all the N_c coefficients, that is, Accuracy$(\mathbf{x}) = (1/N_c)\sum_{i=1}^{N_c}\delta(q_{1,i}(\mathbf{x}), \hat{q}_{1,i}(\mathbf{x}))$, where δ is the Kronecker delta ($\delta(a,b) = 1$ if $a = b$, 0 otherwise). The mean square error (MSE) of the estimation is given by MSE$(\mathbf{x}) = (1/N_c)\sum_{i=1}^{N_c}|q_{1,i}(\mathbf{x}) - \hat{q}_{1,i}(\mathbf{x})|^2$.

The new classification-like structure trained with the loss function $\mathcal{L}$ attempts to maximize the accuracy of the estimation, without caring about the MSE of the estimation. From Eq. (1), it is easy to argue that solutions yielding large MSE values are not penalized compared to those yielding lower values, for the same soft values associated to the "1" positions in vector y ($\mathcal{L}(\mathbf{x})$ takes the same value). Said differently, an incorrect decision on the value of a $q_{1,i}$ for some i, that results in a different wrong one-hot encoded vector, may lead to a same value of the loss function in Eq. (1), regardless of the estimated value, or better yet, regardless of the difference between the true and estimated value, i.e., $|q_{1,i} - \hat{q}_{1,i}|$. Since both the accuracy and the MSE of the estimation are important in practice, we would like to get high accuracy for the estimation, without paying (much) in terms of MSE. In order to solve this issue, we investigated two possible solutions, that correspond to two possible refinements of the loss function. The first solution was to use a "smooth" categorical cross-entropy loss that keeps all the advantages of the standard cross-entropy loss, but at the same time assigns different weights to the errors depending on the position of the "1" inside each one-hot encoded vector, that is, depending on $|q_{1,i} - \hat{q}_{1,i}|$ for each i.

The second solution, that gave us to get better results, was to perform jointly classification and regression by considering a combined loss (as done by some approaches in the literature of deep learning and standard machine learning [15–17]). Given the simil-classification architecture considered, defining a suitable loss function that takes into account the distance between the estimate and the true value, and then penalizes large values of such distance, is not obvious. To do so, we define a vector $\mathbf{d}_y$ that reports in each position the distance from the "1" in the corresponding one-hot encoded vector y_i (see Fig. 2). Formally, let $\mathbf{d}_y = \mathbf{d}_{y,1} \oplus \mathbf{d}_{y,2} \oplus \cdots \oplus \mathbf{d}_{y,N_c}$. The j-th component of $\mathbf{d}_{y,i}$, $i = 1, \cdots N_c$, is given by

$$(d_{y,i})_j = \begin{cases} |j - q_{1,1}| & 1 \le j \le q^M_{1,1} \\ |j - q^M_{1,1} - q_{1,2}| & q^M_{1,1} + 1 \le j \le q^M_{1,1} + q^M_{1,2} \\ \cdots & \cdots \\ \left| j - \left(\sum_{i=1}^{N_c-1} q^M_{1,i}\right) - q_{1,N_c} \right| & \sum_{i=1}^{N_c-1} q^M_{1,i} + 1 \le j \le \sum_{i=1}^{N_c} q^M_{1,i}, \end{cases} \tag{4}$$

where $\mathbf{q}_1$ is the true vector of coefficients.

Then, we define the combined loss function as follows:

$$\mathcal{L}^r(\mathbf{x}) = c \cdot \frac{1}{N_c}\sum_{i=1}^{N_c}\left(\sum_{j\in[1:q^M_{1,i}]} y_{ij}\log(f_{ij}(\mathbf{x}))\right) + + (1-c)\cdot\left(f^T(\mathbf{x})\cdot\mathbf{d}_y\right), \tag{5}$$

where c is a constant, $0 < c < 1$, determining the trade-off between the two terms.

We observe that, for each i, the contribution to the second term is large when the $\arg\max_j f_{ij}(\mathbf{x})$, that is $\hat{q}_{1,i}$, is far from q_{1j}, small otherwise (the second term is 0 when $\arg\max_j f_{ij}(\mathbf{x}) = q_{1i}$ for every i, that is, in the case of ideal estimation). Then, the refined loss indirectly takes into account the MSE of the estimation via the second term.

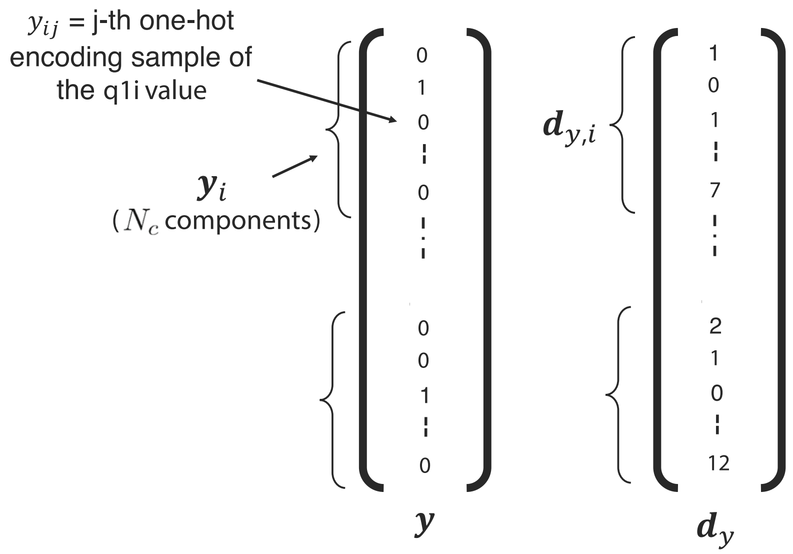

Fig. 2 Vector $\mathbf{d}_y$ of the relative distances obtained from $\mathbf{y}$

Moreover, the second term is continuously differentiable and then is good for backpropagation.

Some preliminary experiments we carried out confirmed that, as expected, adopting the loss function $\mathcal{L}^r$, instead of $\mathcal{L}$, a significantly lower MSE can be reached, at the price of a possible slight decrease in the accuracy. Based on our experiments, by training our CNN model on a mixture of QF_1 ($QF_1 = \{60, 65, 70, 75, 80, 85, 90, 95, 98\}$) with $QF_2 = 90$ for the same number of epochs (100) with the $\mathcal{L}$ and $\mathcal{L}^r$ loss, we got an average accuracy and average MSE of 0.8511 and 1.990 in the first case and 0.8516 and 0.9221 in the second case.

4 Experimental methodology

In this section, we describe in detail the backbone architecture of the network, the procedure of dataset construction and the training setting considered in our experiments.

The proposed solution is compared with the state-of-the-art approach in [10] based on deep learning and, for the aligned case, also with those in [5, 7] based on statistical analysis. While in fact the method in [10] always outperforms all the other previous methods for the non-aligned scenario (e.g., [5, 8]), for the aligned case, there are some cases where the accuracies achieved by the methods in [5, 7], tailored for the aligned scenario, are superior to those of [10].

4.1 Backbone architecture

For the design of the internal layers of the CNN, we considered the DenseNet architecture [14], which was also considered in [10]. Such backbone architecture has been recently adopted for several image forensic tasks, see for instance [18–20], yielding improved performance compared to those achieved with traditional CNN architecture (e.g., residual-based networks).

The main feature of the dense structure is that it connects each layer to every other layer in a dense block in a feed-forward fashion. in this way, the features extracted by the various layers are used by the subsequent layers throughout the same dense block (hierarchical structure). The dense connectivity has been shown in [14] to mitigate the gradient vanishing problem. The number of links in the network increases compared to traditional CNN architectures, passing from l to $l(l-1)/2$ for each dense block, where l is the number of layers in the block. However, as an advantage, the number of (to-be-trained) parameters is significantly reduced. Following the original dense structure (see [14]), we considered a network depth of 40, with 3 dense blocks and growth rate $k = 12$. Each dense layer consists of 12 convolutional layers and a transition layer, where 2×2 average pooling is performed to decrease the input size. All the convolutions have kernel size $3 \times 3 \times 12$. The default dropout of 0.2 is considered. An initial convolution with 24 ($2k$) filters of size 3×3 is performed before the first dense block. For more details on the dense structure, we refer to [14]. After the last dense blocks, global average pooling is performed and the feature vector is fed to the fully connected layer.

The number of output nodes of the fully connected layer is set to $\left(q^M_{1,1} \cdot q^M_{1,2} \cdot \dots \cdot q^M_{1,N_c}\right)$. A softmax is applied to each block of $q^M_{1,i}$ nodes independently, for a total of N_c softmaxes, as illustrated in Fig. 1.

4.2 Datasets

As in [10], a model for Q_1 estimation is trained for a fixed value of QF_2. This does not represent a limitation in practice since the information on the second compression is always available. The knowledge of the final quantization matrix, in fact, can be recovered from the JPEG file, and is necessary to decompress the image, getting the image in the pixel domain. Moreover, when the image is re-saved in an uncompressed format, the quantization matrix of last compression can be accurately estimated [21]. As a drawback, a model has to be trained for every matrix of second quantization, which may be time-consuming (the same happens with the method in [10]). However, since training has to be performed only once, this does not represent a big issue. Moreover, our experiments show that a network trained for a given QF_2 generalizes pretty well to a different QF_2's or, more in general, to a different Q_2 matrix, when the difference is not too much (a ± 2 mismatch in the QF resulting in a very small decrease of performance), hence a limited number of models can be trained. The training and testing datasets are built as described in the following.

We considered the RAISE dataset [22] with 8156 native (tiff) images, that is split into a training and a test set. Specifically, 7000 images were considered for training, while the remaining 1156 were reserved for testing. The images were then compressed first with several QF_1's and then with the prescribed QF_2, thus obtaining several double compressed versions (both QF_1 values larger and smaller than QF_2 were considered). JPEG compression was performed with OpenCV. To simulate the misalignment, we applied a random grid shift (r, c) with $0 \leq r, c \leq 7$ between the two compressions, with r, c randomly selected in the [0:7] range. Therefore, the JPEG is non-aligned with probability 63/64, while the aligned scenario (which corresponds to the case $r = c = 0$) occurs with probability 1/64.

To build the dataset of patches used for training, we proceeded as follows: for every QF_1, we cropped the DJPEG images in the training set into patches of size $64 \times 64 \times 3$; then, from each image, we took 100 patches in random positions; we stopped collecting patches when a total number of 10^5 patches was reached (coming then from 1000 images) for each given QF_1[2]. For our experiments, we set $QF_2 = 90$ and 80. Specifically, for $QF_2 = 90$, we built the training dataset $\mathcal{D}^{90}$ by considering $QF_1 \in [60 : 98]$, for a total of 3.9×10^6 patches. For $QF_2 = 80$, we built set $\mathcal{D}^{80}$ by considering $QF_1 \in [55 : 98]$, then for a total of 4.4×10^6 patches. The test patch set was obtained in the same way. In this case, for every QF_1, all the 1156 images in the test set were considered, each one contributing to 100 random patches (for a total of 115600 patches). By following [10], we directly feed the 3 (R, G and B) channels to the CNN. Another possibility would be to first perform color space conversion from RGB to YC_bC_r and then feed the transformed image to the CNN (since passing from RGB to YC_bC_r corresponds to the application linear mapping, the network should in principle be able to learn the mapping itself, if it benefits the learning process).

The Dresden dataset [23] was also considered to test the performance under dataset mismatch, consisting of 1491 raw images (hence, for each QF_1, the performance were tested on a total of 149,100 patches). The size of the images is around 3600×2700, which is a bit smaller than the size of RAISE images (4928×3264). To further investigate the behavior of our method in presence of a resolution mismatch, we also consider subsampled versions of the raw images from the Dresden database, corresponding to a resolution less than half of the resolution considered for training.

Finally, we also run some tests in the case where the first compression is carried out by using Photoshop (PS), that does not use standard quantization matrices, thus representing a case of strong mismatch between training and testing.

4.3 Setting

In all the experiments, we set $N_c = 15$, which is the value considered in most prior works [5–8, 10]. Hence, we estimate the first 15 DCT coefficients, zig-zag scanned.

For the implementation of the proposed method and the custom loss, we used TensorFlow version 2.2. Model training and testing were carried out in Python via TensorFlow, using Keras API.

We ran our experiments using a 2x Nvidia GeForce RTX 2080 Ti 11 GB GDDR6 GPU. For the optimization, the Adam solver was used with learning rate 10^{-5}. The batch size for training and testing was set to 32 images. We got our models using the $\mathcal{L}^r$ loss by training the network for 100 epochs. After this number of epochs, we verified that the loss decreases very slightly (less than 0.01% at every iterations) and the accuracy of the estimation cannot be improved further by letting the training go on (only incurring the risk of overfitting). The weight in the combined loss $\mathcal{L}^r$ in Eq. (5) was set to $c = 0.8$.

The code is publicly available and can be found at the github link https://tinyurl.com/yxhl32w5.

5 Results

5.1 Test performance and comparison

The average performance results achieved by the CNN model for the new architecture, trained with the $\mathcal{L}^r$ loss on $\mathcal{D}^{90}$, are reported in Table 1, where they are compared to those achieved by the CNN model in [10], trained for the same value of $QF_2 = 90$. The performance results are

[2]For every QF_1, a random shuffle was applied to the 7000 DJPEG images compressed with (QF_1, QF_2), so the subset of images considered for every QF_1 was never the same.

Table 1 Average performance of the proposed CNN estimator and [10] for $QF_2 = 90$

	Prop CNN	**CNN in [10]**
AvgAcc	**0.689**	0.547
AvgMSE	**0.731**	0.882

The DJPEG is non-aligned with probability 63/64, aligned with probability 1/64 (same setting considered for the training of the models)

averaged under the same setting considered for the training regarding the alignment of the DJPEG, i.e., the test patches are DJPEG, aligned with probability 1/64, non-aligned with 63/64. The performance results of the new model are superior to those achieved by the method in [10] both in terms of accuracy and of MSE.

The performance results achieved by the methods in the aligned DJPEG scenario are reported in Table 2. The performance results in aligned scenario are slightly inferior to those reported in the Table 1 for the mixed case. From the results of Table 2, we see that the proposed method can also outperform [7], which is specifically designed for the aligned case, both in terms of MSE and accuracy.

The estimation accuracy and MSE for each DCT coefficient are reported for the non-aligned and aligned case in Figs. 3 and 4, where the results are averaged on all the QF_1s. Comparison with the methods [5, 7] is also reported for the aligned case. It can be observed that the proposed method greatly outperforms these methods, for all the 15 DCT coefficients.

Figure 5 shows the averaged results when $QF_1 < QF_2$ and $QF_1 \geq QF_2$ respectively, in the case QF_2=90 for the aligned scenario. It can be noticed that when $QF_1 \geq QF_2$ the proposed method clearly outperforms the methods [5, 7], both in terms of accuracy and MSE. When $QF_1 < QF_2$, the proposed method still outperforms the state-of-the-art methods.

The average performance obtained for the case $QF_2 = 80$ (model trained on $\mathcal{D}^{80}$) are reported in Table 3 for the non-aligned scenario and Table 4 for the aligned scenario. A performance loss is experienced by all the methods. This was expected since with a smaller QF_2, the second quantization tends to erase more the traces of the first compression, thus making the estimation harder. Nevertheless, the proposed method has an advantage over the state-of-the-art. We see that the CNN model in [10] does better than our method in terms of MSE for the aligned case; however, our method outperforms [10] in terms of accuracy in the aligned case and both in terms of accuracy and MSE in the non-aligned case, which is our main focus. Figures 6 and 7 report the results on 15 DCT coefficients in the case $QF_2 = 80$, averaged on all the QF_1s, for the non-aligned and aligned case respectively. We see that the gain of the proposed method is confirmed.

Table 2 Average performance of the proposed CNN estimator in the aligned case for $QF_2 = 90$, and comparison with the state-of-the-art

	Prop CNN	**CNN in [10]**	**[5]**	**[7]**
AvgAcc	**0.627**	0.463	0.366	0.518
AvgMSE	**1.120**	1.326	28.016	9.091

5.2 Generalization capability

The generalization capability of the model are tested by considering several sources of mismatch, i.e., the second compression quality QF_2, and the image database.

The results in presence of QF_2 mismatch are reported in Fig. 8, where the model trained on $\mathcal{D}^{90}$ is tested on images compressed with $QF_2 = 92$, in the same general setting regarding the alignment considered for training (that is, the DJPEG is aligned with probability 1/64). We see that the drop of performance is limited, proving a certain generalization capability, and is similar for the two methods. The performance of the proposed CNN remains superior to [10]: the total AvgAcc and AvgMSE are respectively 0.591 and 0.859 for our method, and 0.500 and 0.944 for the method in [10]. Notably, in the aligned scenario, the performance remains superior to those of the state-of-the art methods in [5, 7] designed for this case. Specifically, the AvgAcc and AvgMSE are 0.579 and 1.029 for our method, 0.331 and 15.8 for [7], and 0.397 and 35.5 for [5].

To assess the impact that dataset mismatch has on the performance of our CNN-based estimator, we also evaluate the performance of the estimator on DJPEG images coming from the Dresden dataset. Figure 9 reports the results of our tests for $QF_2 = 90$. The total AvgAcc and AvgMSE are respectively 0.644 and 0.538 for our method, and 0.523 and 0.694 for the method in [10]. The results with the half-resolution images from Dresden dataset (strong resolution mismatch) are reported in Fig. 10. As expected, the performance decreases, but not seriously so, and the superior performance of our method over [10] are confirmed also in this case.

Finally, the results of our tests in the case where the first compression is performed with non-standard quantization matrices are provided in Fig. 11, where we report the accuracy and MSE achieved for some common medium-high Photoshop qualities (compression levels from 7 to 12), respectively, for the aligned and non-aligned case.

Table 3 Average performance of the proposed CNN estimator and [10] for $QF_2 = 80$

	Prop CNN	**CNN in [10]**
AvgAcc	**0.440**	0.375
AvgMSE	**1.866**	1.946

The DJPEG is non-aligned with probability 63/64, aligned with probability 1/64 (same setting considered for the training of the models)

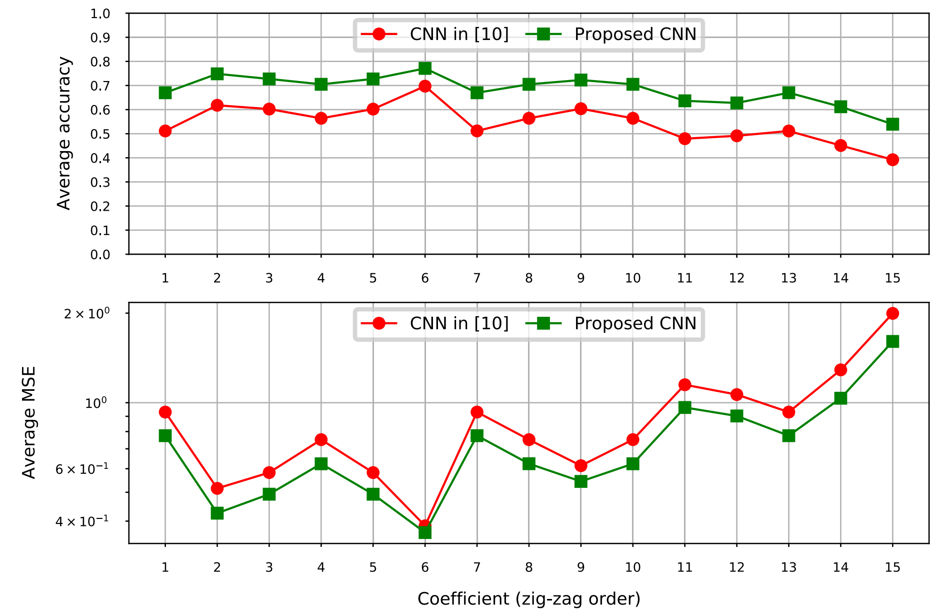

Fig. 3 Average accuracy (top) and average MSE (bottom) of the estimation for each of the 15 DCT coefficients, for $QF_2 = 90$, in the non-aligned DJPEG scenario

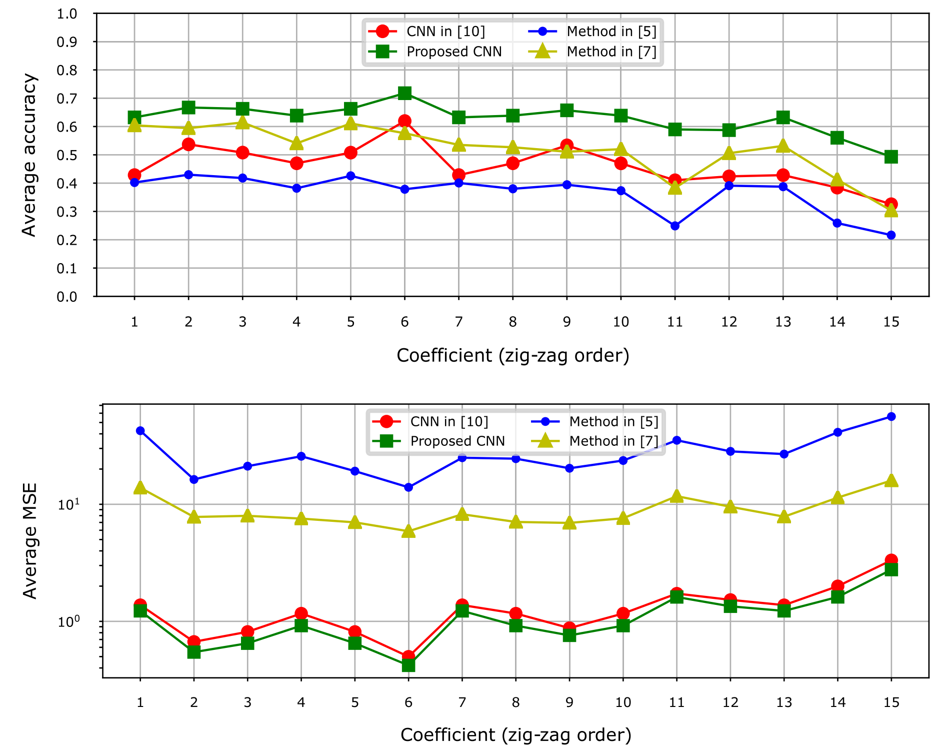

Fig. 4 Average accuracy (top) and average MSE (bottom) of the estimation for each of the 15 DCT coefficients, for $QF_2 = 90$, in aligned DJPEG scenario

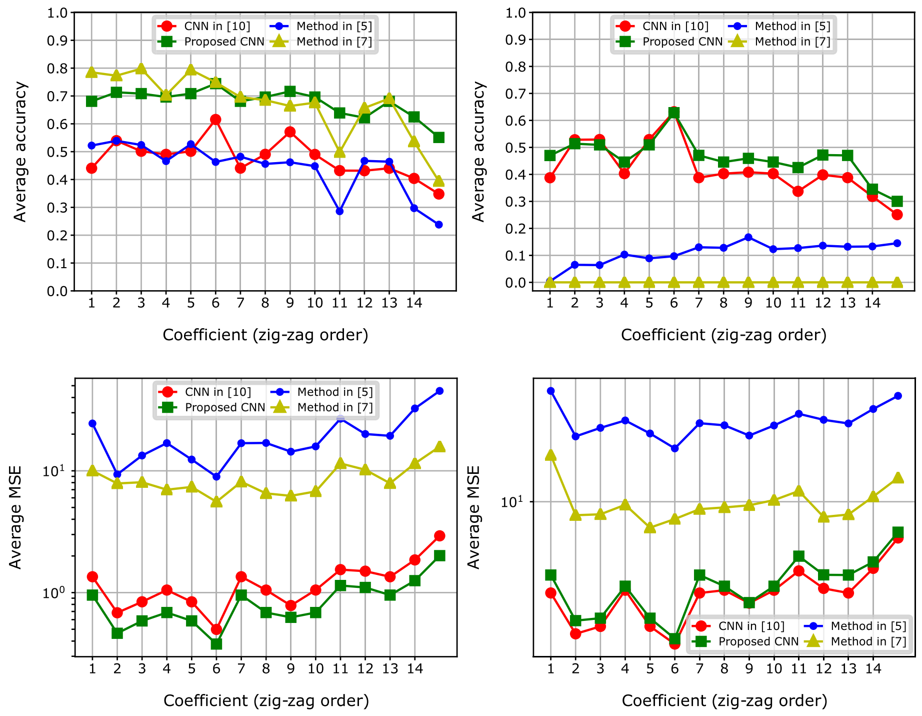

Fig. 5 Average accuracy when $QF_1 < QF_2$ (top left) and when $QF_1 \geq QF_2$ (top right), average MSE when $QF_1 < QF_2$ (bottom left) and when $QF_1 \geq QF_2$ (bottom right) of the estimation for each of the 15 DCT coefficients, for $QF_2 = 90$, in aligned DJPEG scenario

The performance are compared with [10] and with the baseline model-based methods for this case. The performance of [10] and the proposed methods are very similar, our method being superior only slightly on the average and in terms of MSE. For the non-aligned case, both methods significantly outperform the method in [5] and the one in [8] for PS qualities above 8. In the aligned case, instead, the tailored method in [7] works better for low PS qualities, while for higher qualities, the CNN-based estimators gets much better performance (the case where the first quantization step is smaller than the second one is a scenario that model-based techniques cannot handle properly). Clearly, the performance of our CNN estimator for low PS qualities can be improved if the model is trained, or even just fine-tuned, considering examples of JPEG images compressed with non-standard quantization matrices.

Table 4 Average performance of the proposed CNN estimator in the aligned case for $QF_2 = 80$, and comparison with the state-of-the-art

	Prop CNN	**CNN in [10]**	**[5]**	**[7]**
AvgAcc	**0.360**	0.254	0.202	0.282
AvgMSE	5.594	**4.970**	35.453	15.061

5.3 Application to tampering localization

Given the capability of our CNN estimator to work on small image patches, quite straightforwardly, the method can be exploited to localize possible tampering regions in a DJPEG image. Specifically, given a DJPEG tampered image, the estimator can applied on sliding windows to get a map with the estimated primary quantization coefficients $(\hat{\mathbf{q}}_1)_{N_c}$ for each 64×64 block. Notably, by looking at those maps, the tampering can be exposed in the general scenario where both the background and the foreground (tampered areas) are DJPEG, that is, both the background and the copy-pasted region were originally JPEG (compressed with a different qualities) and undergo a second JPEG compression after forging. This corresponds to a very common scenario in practice. In this scenario, methods that try to detect and localize tampering by looking at the presence or absence of typical double compression artifacts do not work, see for instance [3, 24–26], just to mention a few of them. These methods in fact implicitly assume that the background is single compressed, while the foreground is double compressed, or vice versa[3].

[3] As we mentioned in the introduction, for a single compressed region, our estimator returns a quantization matrix of a very high compression quality (to give an insight, the coefficients of the quantization matrix for $QF = 100$ are estimated with an accuracy larger than 0.8). Therefore, when the method is applied for tampering localization in this scenario, the manipulation can still be exposed based on the inconsistencies between the estimated coefficients of background and foreground.

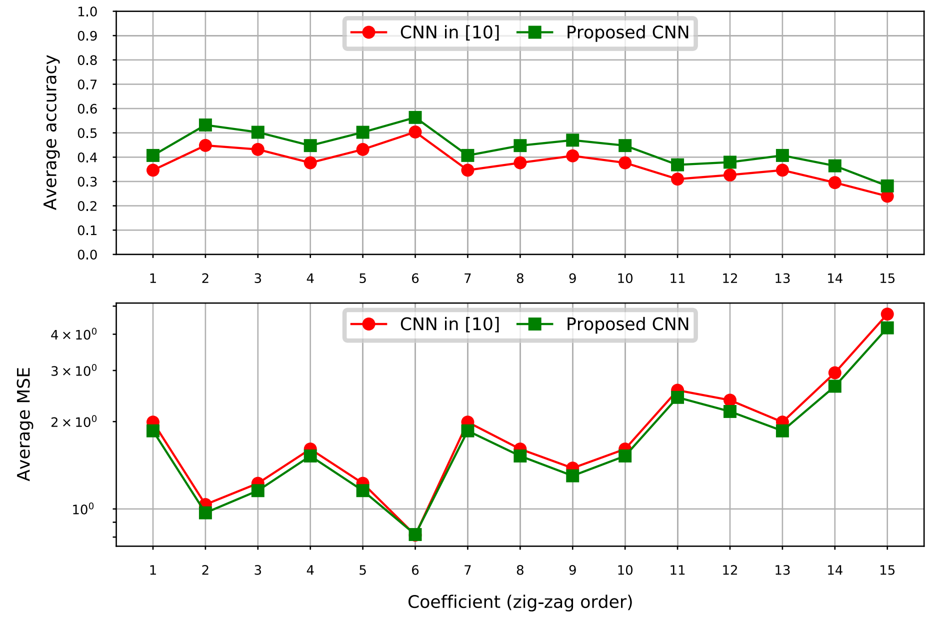

Fig. 6 Average accuracy (top) and MSE (bottom) of the estimation for each of the 15 DCT coefficients, for $QF_2 = 80$, in the non-aligned DJPEG scenario

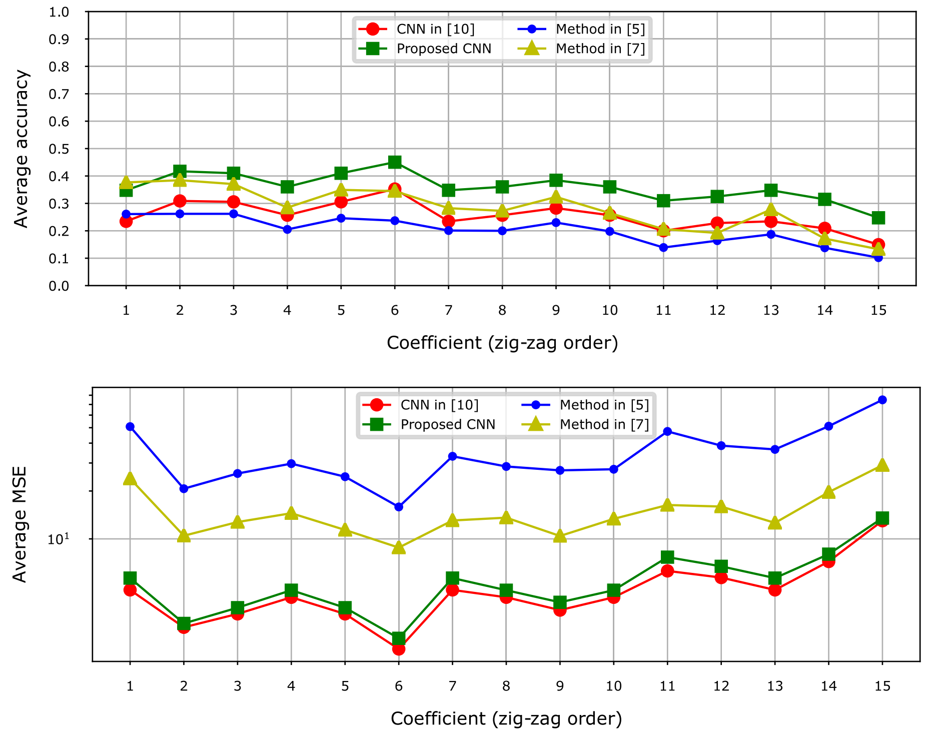

Fig. 7 Average accuracy (top) and MSE (bottom) of the estimation for each of the 15 DCT coefficients, for $QF_2 = 80$, in the aligned DJPEG scenario

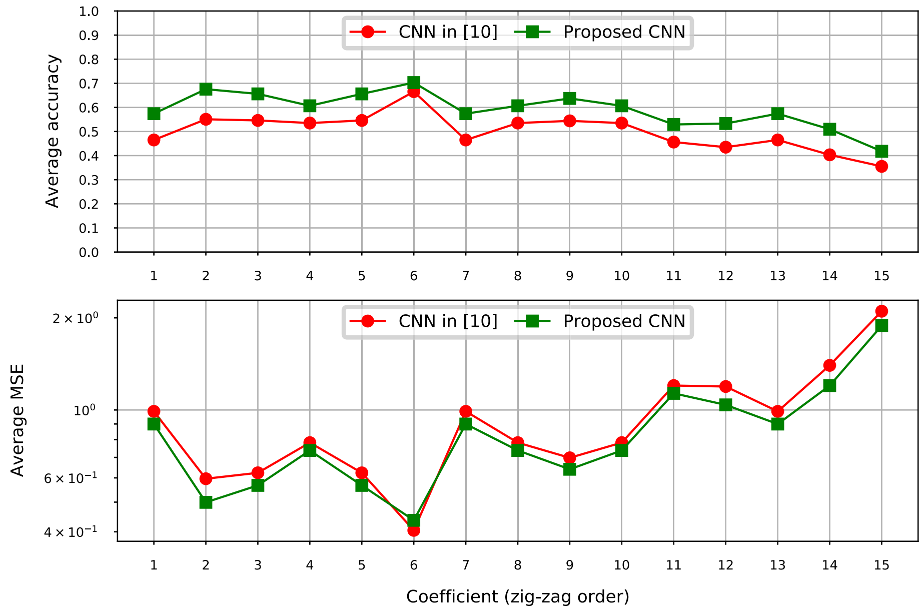

Fig. 8 Performance of the CNN estimator under mismatched QF_2 ($QF_2 = 92$). Average accuracy (top) and MSE (bottom) of the estimation for each of the 15 DCT coefficients

In order to get a localization map, we first divide the input image $\mathbf{x}$ of size $V \times L \times 3$ into overlapping blocks of size 64 ×64 with stride $s = 1$; then, each block is fed to the CNN that returns a vector with the first N_c estimated quantization coefficients. Let $QM(i,j,:) = f([\mathbf{x}_{ij}])$ be the network output when the input is the (i,j)-th image block of size 64 ×64 ×3; then, $QM(i,j,:) = (\hat{\mathbf{q}}_1([\mathbf{x}_{ij}]))_{N_c}$. In this way, for each $k = 1, ..., N_c$ we obtain a map $QM(:,:,k)$ with the estimated values of the k-th coefficient for each block.

We performed a qualitative analysis. Figure 12 shows two examples. For both tampered images, we have two distinct tampered areas, where the copy-pasted regions have different first JPEG qualities, that is, $QF_{1,1}$ = 95 and $QF_{1,2}$ = 85 for the first example and $QF_{1,1}$ = 65 and $QF_{1,2}$ = 95 for the second one. The first JPEG quality for the background of the two examples is 75. The last quality factor for both examples is $QF_2 = 90$. All the JPEG grids are not aligned. For sake of visualization, a color map is reported. The color map shows that the two tampering regions have a different $q_{1,k}$ from the background; interestingly, the color map also reveals that the $q_{1,k}$ value is also different between them, hence, that they correspond to two distinct tampering (the copy-pasted regions come from different donor images, having different JPEG compression qualities).

In general, if the qualities of the former JPEG are close, it is harder to visualize and expose the tampering by simple inspection of the N_c maps of the $q_{1,k}$ coefficients of each block, that is $QM(:,:,k)$, all the more that some of the coefficients might have the same value. In this cases, we could resort to clustering to get a tampering localization map from the vectors of the estimated $q_{1,k}$ values for each position. This interesting analysis is left as a future work.

6 Conclusions

In this paper, we proposed a method for primary quantization matrix estimation via CNNs, that resorts to a classification-like architecture to perform the estimation of the quantization coefficients. Thanks to the adoption of a simil-classification structure, the new CNN estimator achieves improved performance with respect to the CNN regression-based method in [10], both in terms of accuracy and MSE.

Notably, the proposed method is a general one, which can work under a wide variety of operative conditions, i.e., when the second compression grid is either aligned or not with the one of first compression, and for every combinations of qualities of former and second compression. Regarding the JPEG alignment, the method is designed to work in particular for the case of non-aligned double JPEG compression (the aligned case is assumed to occur with probability 1/64). A method capable to deal with primary quantization estimation in the non-aligned scenario, in fact, is very relevant when the proposed estimator is used

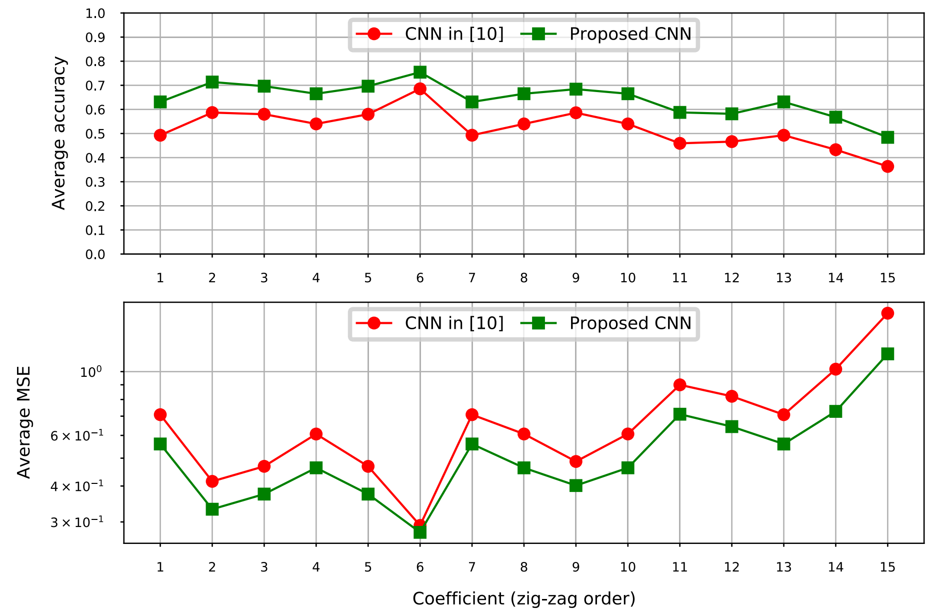

Fig. 9 Performance of the estimator on DJPEG images from a different database (Dresden), for $QF_2 = 90$. Average accuracy (top) and MSE (bottom) of the estimation for each of the 15 DCT coefficients

for image tampering localization (when a region of a JPEG image is copy-pasted into another JPEG image, in fact, very likely, the alignment between the compression grids is not preserved and the final JPEG is non-aligned with the grid of the spliced area). Despite its generality, the proposed method also outperforms the existing — dedicated — state-of-the-art solution for the aligned scenario in most of the cases. The method provide very good performance also in the challenging case of $QF_1 > QF_2$, where state-of-the-art methods based on

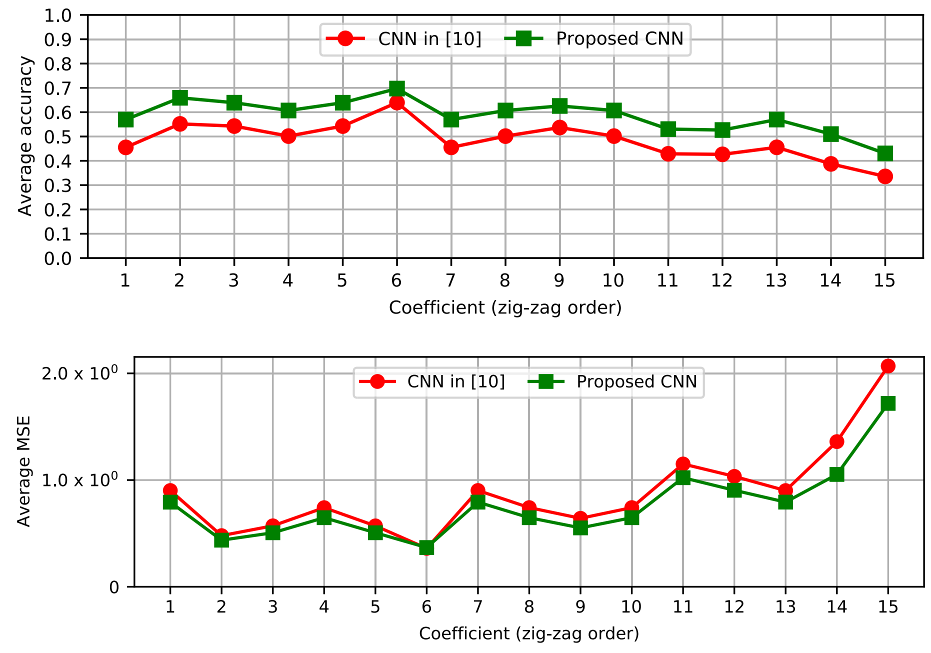

Fig. 10 Performance of the estimator on DJPEG low resolution images from Dresden dataset ($QF_2 = 90$). Average accuracy (top) and MSE (bottom) of the estimation for each of the 15 DCT coefficients

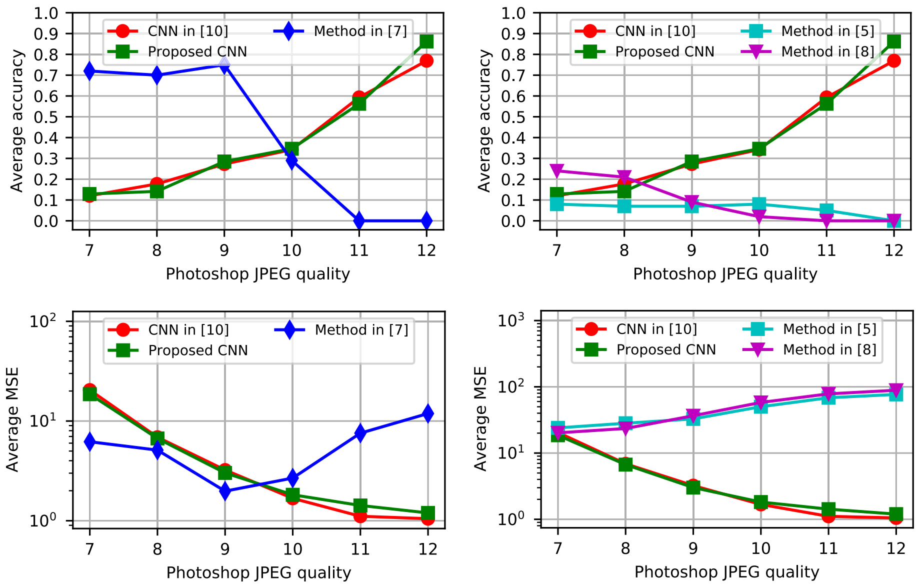

Fig. 11 Performance of the estimator when the first compression is performed with Photoshop for several PS qualities ($QF_2 = 90$), in the aligned (first column) and non-aligned (second column) scenario

statistical analysis often fail. More importantly, the estimator works on small image patches that opens the way to the application of the method for tampering localization (see Section 5.3).

As future research, the application of the method for tampering localization in DJPEG images, possibly including the identification of the different tampering sources (donors) is an interesting direction. In addition, the robustness of the estimator in presence of adversarial attacks could also be investigated. From a more general perspective, we believe that a similar architecture to the one proposed in this paper could also be exploited to address other estimation problems which are relevant in image forensics.

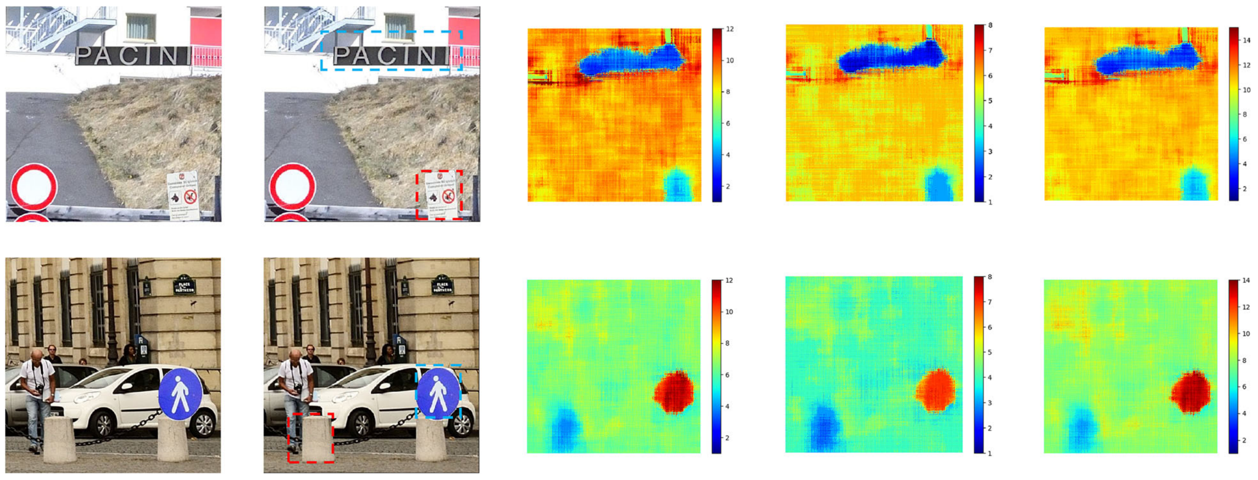

Fig. 12 Examples of tampered (double JPEG) images and map of the estimated k-th Q_1 coefficient, for some values of k. For each example, we report, from left to right: the tampered image, the ground-truth tampering map, the estimated map for the DCT coefficient no 1, 6, and 14, that is $QM(:,:,1)$, $QM(:,:,6)$ and $QM(:,:,14)$

Abbreviations

JPEG: Joint photographic experts group; DJPEG: Double JPEG; DCT: Discrete cosine transform; QF: Quality factor; CNN: Convolutional neural network; MSE: Mean squared error; AvgACC: Average accuracy; AvgMSE: Average mean squared error; API: Application programming interface

Authors' contributions

BT conceived the presented idea and developed the theoretical formalism. BL supervised the study and helped interpreting the results. BT wrote the paper, with the support of BL. AC and BT conceived and planned the experiments. AC implemented the method and helped interpreting the results. DH and AC carried out the experiments and positioned the work in the current state of research. The authors read and approved the final manuscript.

Funding

This work has been partially supported by the National Natural Science Foundation of China (Grant 61872244), Guangdong Basic and Applied Basic Research Foundation (Grant 2019B151502001), Shenzhen R&D Program (Grant JCYJ20200109105008228, JCYJ20180305124325555), and by the Italian Ministry of University and Research (MUR) under the PRIN 2017 2017Z595XS-001 program — PREMIER project.

Availability of data and materials

The datasets used for the experiments are publicly available in [22] and [23]. The code is available at the following Github link: https://github.com/andreacos/BoostingCNN-Jpeg-Primary-Quantization-Matrix-Estimation.

Declarations

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Department of Information Engineering and Mathematics, University of Siena, via Roma 56, 53100 Siena, Italy. [2]Guangdong Key Laboratory of Intelligent Information Processing and Shenzhen Key Laboratory of Media Security, Shenzhen University, No 3688 Nanhai Avenue, 518060 Shenzhen, China.

Received: 9 October 2020 Accepted: 21 April 2021
Published online: 17 May 2021

References

1. T. Pevny, J. Fridrich, Detection of double-compression in JPEG images for applications in steganography. IEEE Trans. Inf. Forensics Secur. **3**(2), 247–258 (2008). https://doi.org/10.1109/TIFS.2008.922456
2. B. Li, Y. Q. Shi, J. Huang, in *2008 IEEE 10th Workshop on Multimedia Signal Processing*. Detecting doubly compressed JPEG images by using mode based first digit features, (2008), pp. 730–735. https://doi.org/10.1109/MMSP.2008.4665171
3. M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, S. Tubaro, Aligned and non-aligned double JPEG detection using convolutional neural networks. J. Vis. Commun. Image Represent. **49**, 153–163 (2017). https://doi.org/10.1016/j.jvcir.2017.09.003
4. W. B. Pennebaker, J. L. Mitchell, *JPEG: still image data compression standard*. (Springer, New York, 1992)
5. T. Bianchi, A. Piva, Image forgery localization via block-grained analysis of JPEG artifacts. IEEE Trans. Inf. Forensics Secur. **7**(3), 1003–1017 (2012). https://doi.org/10.1109/TIFS.2012.2187516
6. T. H. Thai, R. Cogranne, Estimation of primary quantization steps in double-compressed JPEG images using a statistical model of discrete cosine transform. IEEE Access. **7**, 76203–76216 (2019). https://doi.org/10.1109/ACCESS.2019.2921324
7. F. Galvan, G. Puglisi, A. R. Bruna, S. Battiato, First quantization matrix estimation from double compressed JPEG images. IEEE Trans. Inf. Forensics Secur. **9**(8), 1299–1310 (2014). https://doi.org/10.1109/TIFS.2014.2330312
8. N. Dalmia, M. Okade, Robust first quantization matrix estimation based on filtering of recompression artifacts for non-aligned double compressed JPEG images. Signal Process. Image Commun. **61**, 9–20 (2018). https://doi.org/10.1016/j.image.2017.10.011
9. S. Battiato, O. Giudice, F. Guarnera, G. Puglisi, *In-depth DCT coefficient distribution analysis for first quantization Estimation*. (Springer International Publishing, 2021), pp. 573–587. https://doi.org/10.1007/978-3-030-68780-9_45, http://arxiv.org/abs/2008.03206
10. Y. Niu, B. Tondi, Y. Zhao, M. Barni, Primary quantization matrix estimation of double compressed JPEG images via CNN. IEEE Signal Process. Lett. **27**, 191–195 (2020). https://doi.org/10.1109/LSP.2019.2962997
11. A. Bulat, G. Tzimiropoulos, in *European Conference on Computer Vision*. Human pose estimation via convolutional part heatmap regression (Springer, 2016), pp. 717–732
12. L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
13. R. Alp Guler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, I. Kokkinos, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Densereg: fully convolutional dense shape regression in-the-wild, (2017), pp. 6799–6808
14. G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Densely connected convolutional networks, (2017), pp. 4700–4708
15. A. Kendall, Y. Gal, R. Cipolla, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, (2018), pp. 7482–7491
16. M. Liu, J. Zhang, E. Adeli, D. Shen, in *Medical Image Computing and Computer Assisted Intervention MICCAI 2017*, ed. by M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, and S. Duchesne. Deep multi-task multi-channel learning for joint classification and regression of brain status (Springer, Cham, 2017), pp. 3–11
17. X. Zhu, H.-I. Suk, D. Shen, A novel matrix-similarity based loss function for joint regression and classification in ad diagnosis. NeuroImage. **100**, 91–105 (2014)
18. Y. Chen, X. Kang, Y. Q. Shi, Z. J. Wang, A multi-purpose image forensic method using densely connected convolutional neural networks. J. Real-Time Image Proc. **16**(3), 725–740 (2019)
19. U. Kamal, A. M. Rafi, R. Hoque, S. Das, A. Abrar, M. Hasan, *et al*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. Application of densenet in camera model identification and post-processing detection, (2019), pp. 19–28
20. J. Yang, J. Xie, G. Zhu, S. Kwong, Y. Q. Shi, An effective method for detecting double JPEG compression with the same quantization matrix. IEEE Trans. Inf. Forensics Secur. **9**(11), 1933–1942 (2014)
21. P. Bestagini, A. Allam, S. Milani, M. Tagliasacchi, S. Tubaro, in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference On*. Video codec identification (IEEE, 2012), pp. 2257–2260
22. D. Dang-Nguyen, C. Pasquini, V. Conotter, G. Boato, in *Proceedings of the 6th ACM Multimedia Systems Conference, MMSys '15,*. RAISE: a raw images dataset for digital image forensics (ACM, New York, NY, USA, 2015), pp. 219–224. https://doi.org/10.1145/2713168.2713194
23. T. Gloe, R. Böhme, in *Proceedings of the 2010 ACM Symposium on Applied Computing, SAC '10*. The 'Dresden Image Database' for benchmarking digital image forensics (ACM, New York, NY, USA, 2010), pp. 1584–1590. https://doi.org/10.1145/1774088.1774427
24. T. Bianchi, A. De Rosa, A. Piva, in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Improved DCT coefficient analysis for forgery localization in JPEG images (IEEE, 2011), pp. 2444–2447
25. I. Amerini, R. Becarelli, R. Caldelli, A. Del Mastio, in *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. Splicing forgeries localization through the use of first digit features (IEEE, 2014), pp. 143–148
26. Q. Wang, R. Zhang, Double jpeg compression forensics based on a convolutional neural network. EURASIP J. Inf. Secur. **2016**(1), 23 (2016)