Robust High-Capacity Watermarking over Online Social Network Shared Images

Weiwei Sun, *Student Member, IEEE,* Jiantao Zhou, *Senior Member, IEEE,* Yuanman Li, *Member, IEEE,* Ming Cheung, *Member, IEEE,* and James She, *Member, IEEE*

***Abstract*—In recent years, online social networks (OSNs) have become extremely popular and been one of the most common ways for storing and distributing images. Naturally, such widespread availability of OSN makes it a viable channel for transmitting additional data along with the image sharing. However, various lossy operations, e.g., resizing and compression, conducted by OSN platforms impose great challenges for designing a robust watermarking scheme over OSN shared images. In this paper, we tackle this challenge and propose a robust high-capacity watermarking technique, by using Facebook as a representative OSN. To achieve the satisfactory robustness, we first probe into Facebook and recover the image manipulation mechanism via a deep convolutional neural network (DCNN) approach. Assisted with the precise knowledge on the lossy channel offered by Facebook, we then suggest a DCT-domain image watermarking method that is highly robust against the lossy operations on Facebook, even without any error correcting codes (ECC). The proposed technique is also extended to other popular OSNs, e.g., Wechat and Twitter. Extensive experimental results are provided to show the superior performance of our method in terms of the embedding capacity, data extraction accuracy, and quality of the reconstructed images.**

***Index Terms*—Image watermarking, Online social networks, Deep convolutional neural network, DCT domain**

I. Introduction

Nowadays, the popularity of various online social network (OSN) platforms significantly simplifies the dissemination and sharing of images. A huge number of images have been uploaded to Facebook, Twitter, and Wechat, etc. at a daily basis. For instance, there are over 350 million new photos uploaded to Facebook [1] everyday. For Wechat, this number goes to astoundingly 1 billion [2]. Obviously, OSNs have become one of the most accessible ways for storing and sharing images, making OSN platform a practically available channel for transmitting additional data along with the image sharing. However, designing a feasible watermarking scheme over OSN shared images is quite challenging, primarily due to the following two reasons. First of all, as indicated by many existing works [3]–[5], almost all OSNs manipulate the uploaded images in a lossy fashion, which could severely affect the extraction of the embedded data. For instance, Facebook converts all uploaded images to the pixel domain with rounding and possible resizing. JPEG compression is then applied with quality factor (QF) adaptively chosen. In other words, the links provided by OSNs cannot be regarded as perfect channels, but rather lossy ones. Secondly, these lossy operations conducted by OSNs are typically unknown, and beyond the controllable domain of users.

The most relevant technique for embedding data over OSN shared images is the traditional robust image watermarking, which has been extensively investigated for more than two decades [6]–[32]. Most of the existing robust image watermarking methods were usually designed to be robust against one or several hypothetical image manipulations, e.g., image cropping, scaling, or JPEG compression, which cannot fully reflect the characteristics of the real lossy channel in practice. Moreover, these robust watermarking methods typically can only accommodate very limited number of embedded bits, making them far from adequate for use in an enormously complex and lossy scenario. In recent years, some approaches specifically attempted to design robust image watermarking over OSNs. Ning *et al.* suggested to use the 2nd-LSBs of the quantized DCT coefficients for robust data embedding over various OSN platforms [4]. However, the extraction accuracy of the embedded data is very low and even approaches 50% when applied to Facebook. Morkel *et. al.* [33] then introduced a tagging watermarking algorithm, in which the embedding is achieved by modulating all the pixels according to the bits to be embedded. Nevertheless, the high extraction accuracy is achieved with the available original image, also at the cost of severe blocking artifacts. In addition, the embedding capacity is generally low for this method, as only one bit can be embedded into each block. To our best knowledge, [4] and [33] are the only two existing works which investigated the robust watermarking over OSN shared images. In [34], Tao *et al.* designed a robust image steganography method by improving two JPEG steganographic schemes [35], [36]. It was claimed that such method could be applied to OSNs, while all the experiments were conduced in a simulated (offline) environment. More importantly, the practical OSN platforms are much more complicated than a standalone JPEG

This work was supported by Macau Science and Technology Development Fund under SKL-IOTSC-2018-2020, 077/2018/A2, 022/2017/A1 and 0015/2019/AKP, by Research Committee at University of Macau under MYRG2018-00029-FST and MYRG2019-00023-FST, by Natural Science Foundation of China under 61971476, and by Guangdong Basic and Applied Basic Research Foundation under 2019A1515110410. (Corresponding author: Jiantao Zhou, email: jtzhou@um.edu.mo)

W. W. Sun and J. T. Zhou are with the State Key Laboratory of Internet of Things for Smart City, and also with the Department of Computer and Information Science, University of Macau. e-mails: yb57457@um.edu.mo, jtzhou@um.edu.mo.

Y. M. Li is with the College of Electronics and Information Engineering, Shenzhen University. e-mail: yuanmanli@szu.edu.cn.

M. Cheung is with the Social Face Limited. e-mail: ming@socialface.ai.

J. She is with the HKUST-NIE Social Media Lab. e-mail:eejames@ust.hk.

compression module with a fixed QF, as considered in [34].

In this work, we address the challenging task of designing and implementing a robust high-capacity image watermarking scheme over OSNs. We first consider Facebook, one of the most representative OSNs, as the design platform. We first conduct an in-depth investigation on the manipulations that Facebook performs to the uploaded images. This is equivalent to performing the estimation of the lossy channel provided by Facebook. We identify that four types of operations are applied to the uploaded images, including format conversion, resizing, JPEG compression, and enhancement filtering. Except the enhancement filtering, the parameters employed in the other three operations can be *accurately* estimated through an off-line training procedure. Due to the difficulty in modeling the enhancement filtering analytically, we adopt a deep convolutional neural network (DCNN) to predict its effect on the uploaded images. Being aware of the knowledge on the manipulations, we design a DCT-domain high-capacity image watermarking scheme that is robust against these lossy operations. The marked image can be freely spread over Facebook. For an image of size 512×512, the average number of bits that can be embedded is around $23\ K$ bits, which is quite sufficient for many practical applications. At the recipient side, it can be shown that, due to the robust design, the data extraction can be almost perfect ($> 99.99\%$), even without any error correcting codes (ECC). In addition, the original cover image can be recovered with high fidelity upon the data extraction. Extensive experimental results are provided to validate the superior performance of our method in terms of the embedding capacity, data extraction accuracy, and quality of the reconstructed images, compared with the state-of-the-art competing schemes. Furthermore, we extend the developed technique to the other OSN platforms such as Wechat and Twitter, achieving promising performance.

The rest of this paper is organized as follows. Section II presents the system model and design goals of our watermarking scheme over Facebook shared images. In Section III, we describe our findings on how Facebook manipulates the uploaded images, based on which we in Section IV, propose a DCNN approach for selecting the blocks that are immune from the lossy operations conducted by Facebook. Section V is devoted to the proposed robust image watermarking scheme, including the details on the data embedding, the data extraction, and the necessary error analysis. In Section VI, we explain how to extend our watermarking scheme to other OSN platforms. Experimental results are provided in Section VII to demonstrate the superior performance of our scheme. We finally conclude in Section VIII.

II. System Model and Design Goals

The framework of the proposed robust watermarking scheme over Facebook shared images is illustrated in Fig. 1. Before sharing images through Facebook, Alice embeds a message $\mathbf{M}$ into the cover image $\mathbf{I}$. Alice then uploads the marked image $\mathbf{I}'$ to Facebook, exploiting Facebook as a viable channel for transmitting additional information along with the cover image $\mathbf{I}$. As explained in Section I, Facebook applies various lossy operations, including JPEG compression and enhancement filtering, to $\mathbf{I}'$ (details will be given in the next section). The resulting image $\bar{\mathbf{I}}$ of JPEG format is stored in Facebook for sharing purpose. At the client side, Bob downloads $\bar{\mathbf{I}}$ from Facebook and applies the extraction function to extract the message $\mathbf{M}'$ and recover the original image $\hat{\mathbf{I}}$.

To design a robust high-capacity image watermarking scheme over such a highly lossy channel provided by Facebook, the key components are the embedding and extraction functions. The design goals can be summarized as follows.

- ***High quality of the shared image***: The image $\bar{\mathbf{I}}$ stored and shared over Facebook should be of high quality, which is a fundamental requirement of the image sharing.
- ***High embedding capacity***: The embedding capacity should be as high as possible, which has always been an essential indicator for evaluating a watermarking method.
- ***High quality of the extracted data***: The distortion between the extracted message $\mathbf{M}'$ and the embedded $\mathbf{M}$ should be as small as possible. In other words, the embedded data should be able to survive from the highly lossy operations conducted by Facebook. It should be emphasized that we do *not* rely on any ECC for offering the robustness; but rather exploiting the robust property inherent to the image signal itself. As can be seen later, the message extraction is still nearly perfect, even without any ECC.
- ***High quality of the reconstructed image***: The reconstructed image $\hat{\mathbf{I}}$ upon the message extraction should be of high quality as well.

Bearing the above design goals in mind, the most challenging task now is to combat with the negative effect brought by the lossy operations on Facebook, especially noticing that these lossy operations are applied automatically and are beyond the control of users.

III. Detecting the Image Manipulations on Facebook

In order to design a robust image watermarking scheme over Facebook shared images, the first challenge is to figure out the image manipulations on Facebook. It is well-known that almost all the existing OSN platforms apply various operations, e.g., format conversion, compression, enhancement filtering, etc. on user-uploaded images [3]–[5], [37]. These operations, which are lossy in nature, are usually designed to save the storage cost, improve the viewing experiences, or simplify the file management. In other words, the uploaded images could be severely disturbed. As mentioned in [4], [5], [37], the manipulations on Facebook are much more complicated than the other OSNs, and the results obtained over Facebook could be easily extended to the other platforms. That is one of the reasons why we choose Facebook as the representative OSN, in addition to its extreme popularity. In our previous works [37], we have investigated how Facebook processes the uploaded images to some extent, though some operations remain mysterious. The general framework regarding the manipulations performed by Facebook can be illustrated in Fig. 2,

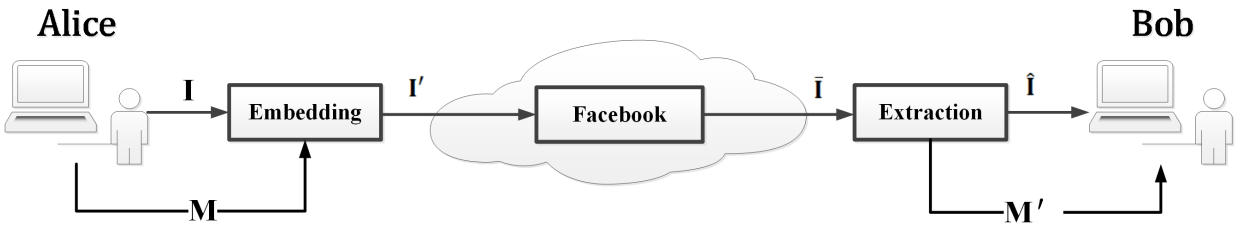


Fig. 1. Overview of the proposed image watermarking scheme over Facebook shared images.

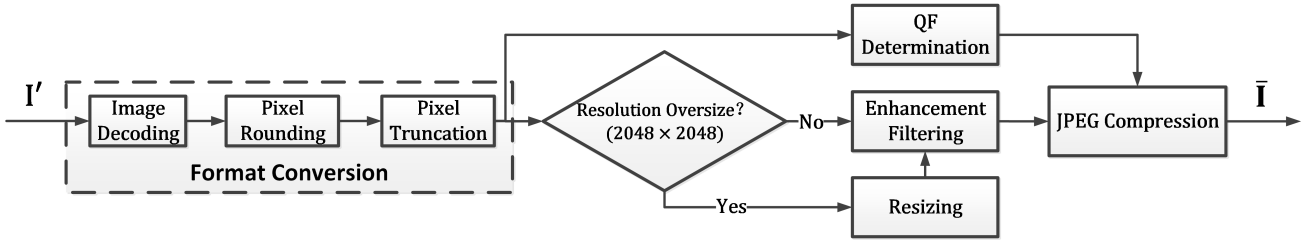


Fig. 2. Manipulations of uploaded images on Facebook.

which consists of four types of operations: format conversion, resizing, JPEG compression, and enhancement filtering.

More specifically, the uploaded images are first decoded into the pixel domain, where the truncation is also applied such that the pixel values are within the range $[0, 255]$. After that, Facebook checks the resolution of the image in the pixel domain, and employs resizing if either the height or the width is greater than 2048 pixels. Subsequently, the resulting image is subject to a round of JPEG compression with QF adaptively determined according to the image content. Through extensive experiments, we have known that the QF used by Facebook ranges from 71 to 95 [5]. In addition to the results achieved in our previous works [5], we find that the QF values used are also closely related to the quality of the uploaded image. Generally, for the image of high quality, the QF value used in Facebook tends to be high, while for the one with low quality, the associated QF value seems to be low. This maybe because the low-quality image is more noisy and more difficult to be compressed; using a low QF could better save the storage cost. To further explore the relationship between the quality of the uploaded image and the QF actually used on Facebook, we assume that the uploaded image is of JPEG format as well. As can be seen from Fig. 3, Facebook adopts a rather conservative strategy: when the QF value of the uploaded image is low, Facebook uses the $QF = 71$, the minimum one in the range $[71, 95]$, to compress it. Only when the quality of the uploaded image is sufficiently high, Facebook then utilizes a QF value higher than 71. It should be noted that this phenomenon is observed not only from these test images in Fig. 3, but also from a much larger dataset. Another important conclusion that can be drawn from Fig. 3: only when the QF value of the uploaded image is 71, the QF value employed by Facebook remains the same, which is valid for *all* the images. As will be clear in the following sections, the QF assignment strategy is very important to design a robust high-capacity image watermarking scheme over Facebook shared images.

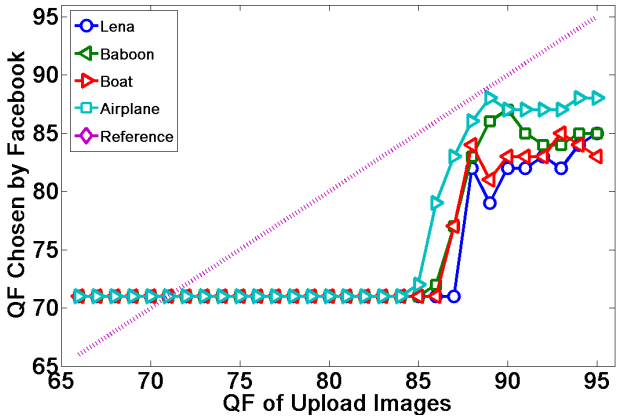


Fig. 3. Relationship between the QF of the upload images and the QF employed by Facebook.

Apart from the aforementioned operations, additional enhancement filtering is also applied to the uploaded image. As mentioned in [5], these operations are highly adaptive and only applied to some selected image blocks, making it very challenging to be determined by traditional signal processing approaches. Although the changes due to the enhancement filtering are small, these changes could lead to the de-synchronization problem in the message extraction phase, causing errors that are difficult to be corrected. In this work, we resolve this challenge by using a deep learning approach. Instead of fully characterizing such enhancement filtering operations, we use a DCNN to identify which blocks remain unchanged after the enhancement filtering, thus converting a complicated regression problem into a much simpler classification problem. The detailed description of this DCNN module is deferred to Section IV.

Upon the clarification of the above operations, let us now analyze the end-to-end distortion caused by these lossy operations. Let $\mathbb{B}' = \{\mathbf{B}'_i\}_{i=1}^{N}$ and $\bar{\mathbb{B}} = \{\bar{\mathbf{B}}_i\}_{i=1}^{N}$ be the collections

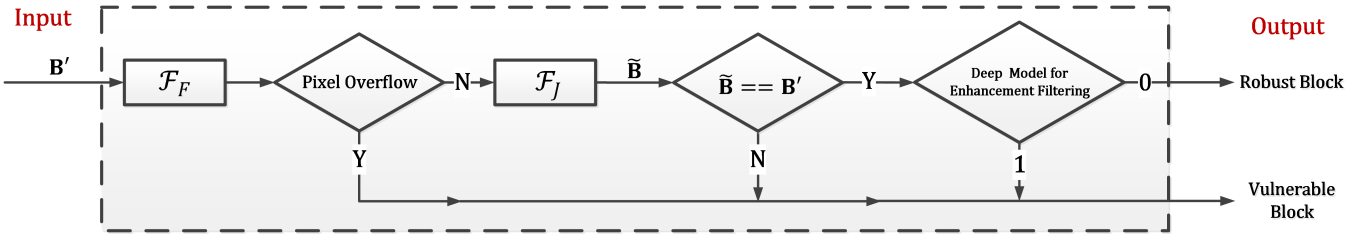


Fig. 4. Structure of the proposed block classifier.

of DCT blocks (8×8) of the image $\mathbf{I}'$ and $\bar{\mathbf{I}}$, respectively, where N is the total number of blocks. Then,

$$d_i = \left\| \bar{\mathbf{B}}_i - {\mathbf{B}'}_i \right\|_2, \tag{1}$$

measures the distortion associated with the ith block due to the lossy operations of Facebook. A conservative yet effective strategy for robust watermarking is to use those blocks ${\mathbf{B}'}_i$ with the associated $d_i = 0$ (zero distortion) for carrying the message bits, while leaving the other blocks untouched. Noticing that we enforce both dimensions of $\mathbf{I}'$ to be no larger than 2048 pixels (no resizing is triggered), we can mathematically model the channel offered by Facebook as

$$\bar{\mathbf{B}}_i = \mathcal{F}_E(\mathcal{F}_J(\mathcal{F}_F({\mathbf{B}'}_i))). \tag{2}$$

where $\mathcal{F}_E(\cdot)$, $\mathcal{F}_J(\cdot)$ and $\mathcal{F}_F(\cdot)$ represent the unknown enhancement filtering, the JPEG compression and the format conversion, respectively. Consequently, (1) can be re-written as

$$d_i = \left\| \mathcal{F}_E(\mathcal{F}_J(\mathcal{F}_F({\mathbf{B}'}_i))) - {\mathbf{B}'}_i \right\|_2. \tag{3}$$

IV. DCNN-Based Robust Block Selection

Based on the knowledge of the lossy operations conducted by Facebook, we in this section develop a DCNN-based classifier to determine whether an image block ${\mathbf{B}'}_i$ keeps unchanged or not after these lossy operations. For a block ${\mathbf{B}'}_i$, if its associated distortion $d_i = 0$ ($d_i \neq 0$), then it is called a robust (vulnerable) block. More specifically, we assign a label v_i to the block ${\mathbf{B}'}_i$, i.e.,

$$v_i = \mathcal{C}({\mathbf{B}'}_i) = \begin{cases} 1 & \text{if } d_i > 0 \\ 0 & \text{if } d_i = 0. \end{cases} \tag{4}$$

For a block $\mathbf{B}'$ with embedded data, if it is identified as a robust block, then the embedding task is claimed to be successful. Otherwise, if it is classified as a vulnerable one, then the embedding operation on it will be cancelled. As will be clear soon, by appropriate robust design, the number of robust blocks is quite large, potentially enabling a high-capacity watermarking scheme, even under this rather conservative strategy. The overall procedure of performing this classification is illustrated in Fig. 4, where the input $\mathbf{B}'$ is a generic quantized DCT block (the subscript i is removed), and the output of the classifier is the binary label v indicating whether it is a robust or vulnerable block. The first building module in Fig. 4 mimics the format conversion for restoring a pixel block, and the QF value used is extracted from the header file associated with $\mathbf{B}'$. If the pixel overflow phenomenon is observed, then the label v is immediately set to be 1, i.e., $\mathbf{B}'$ is classified as a vulnerable block. If otherwise $\mathbf{B}'$ passes this check, it will be applied a round of JPEG compression to produce a new quantized DCT block $\tilde{\mathbf{B}}$, where the QF value is selected according to the assignment strategy on Facebook. Then, $\tilde{\mathbf{B}}$ will be checked with $\mathbf{B}'$. If there is any inconsistency, the label v is set to be 1. Otherwise, the quantized DCT block $\tilde{\mathbf{B}}$ will be further checked by a DCNN-based classifier, corresponding to the unknown enhancement filtering.

Recall that we only need to determine whether the block $\mathbf{B}'$ stays unchanged or not upon the enhancement filtering. Therefore, we do not need to fully regress how the enhancement filter works; but rather a binary classifier is sufficient. To this end, we design a DCNN-based classifier, as illustrated in Fig. 5. This classifier is trained with a set of training blocks $\{\tilde{\mathbf{B}}_i\}_{i=1}^K$, which are quantized DCT blocks with $QF = 71$. It should be emphasized that we do not absorb the operations of $\mathcal{F}_F$ and $\mathcal{F}_J$ into this deep learning based classifier. This is because both $\mathcal{F}_F$ and $\mathcal{F}_J$ can be *accurately* determined; mixing them into the deep learning based classifier could unnecessarily complicate the design and even degrade the classification performance. The developed DCNN is designed to minimize the following cross entropy loss function

$$\mathcal{L}(\Theta) = \frac{1}{N} \sum_{i=1}^{K} \sum_{k=1}^{2} y_i^{*(k)} \log(y_i^{(k)}), \tag{5}$$

where Θ denotes the trainable parameters, $y_i^{*(k)}$ represents the true label (one-hot vector) corresponding to the DCT block $\tilde{\mathbf{B}}_i$, and $y_i^{(k)}$ is the output of the deep learning based classifier for $\tilde{\mathbf{B}}_i$. Fig. 5 shows the architecture of the proposed DCNN, consisting of four types of layers marked in different colors.

- **CNN(3,3)+ReLU**: we use 32 filters of size $3 \times 3 \times 1$ for the first layer to generate 32 feature maps, concatenating rectified linear units (ReLU) [38] for nonlinearity. To ensure that each feature map of the middle layers has the same size as the input, we use zero padding for all the convolution operations.
- **CNN(3,3)+BN+ReLU**: for layers 2∼4, 32 filters of size $3 \times 3 \times 32$ are adopted. Batch normalization [39] is added between the convolution and ReLU to reduce the effect of internal covariate shift. Note that the size of the input is 8×8, and hence 4 convolution layers of $3 \times 3 \times 32$ filter size have enough receptive field (local region/patch convolved with a filter) to cover the entire input.

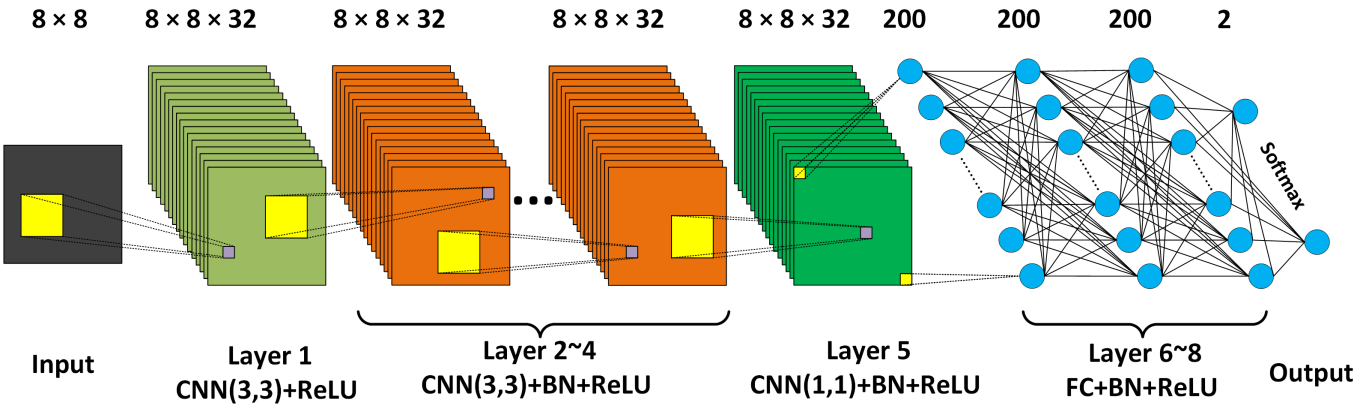


Fig. 5. Overall framework of our proposed DCNN architecture.

- **CNN(1,1)+BN+ReLU**: for the 5th layer, we adopt 32 filters of size $1 \times 1 \times 32$ to constrain the DCNN to learn the association across feature maps produced by the former layers. This technique is widely employed in many DCNN architectures to improve their learning abilities [40].
- **FC+BN+ReLU**: the deepest layers consist of four fully connected layers, each of which has 200 neurons. The outputs of the final fully connected layer will directly be fed into a softmax layer with 2 neurons for the classification.

To train our DCNN, we first build a dataset with 1000 images of size 768×768. We randomly select 80% of them for training, while the remaining for testing. Since we cannot get the intermediate results of the image manipulations on Facebook, the input blocks should be generated at the client side. The format conversion and JPEG compression which mimic the Facebook image manipulations are applied to the training images to obtain the intermediate quantized DCT blocks $\{\tilde{\mathbf{B}}_i\}_{i=1}^{K}$. The QF values used in the JPEG compression are consistently 71. Then, we upload the training images to Facebook and then download them. By entropy decoding the downloaded JPEG files and comparing with $\{\tilde{\mathbf{B}}_i\}_{i=1}^{K}$, we can obtain the corresponding output labels. We observe that, for each image, the average number of blocks that are labeled as 1 (positive sample) accounts for approximately 1.2% of the total number of blocks, i.e., 110 samples per image on average. Therefore, to balance the proportion of the positive and negative samples, the number of negative samples is set to 1.5 times of that of positive samples. Such negative samples are randomly selected from all the blocks which are labeled as 0. Finally, we totally have 220 K quantized DCT blocks and the corresponding labels $\{\tilde{\mathbf{B}}_i, y_i^*\}$ for training, and 55 K pairs for testing.

The network parameters are initialized using Xavier method [41]. We use stochastic gradient descent (SGD) to train our network with a weight decay of 10^{-4}, a momentum of 0.9, and the mini-batch size of 96. The DCNN is trained for 100 epochs with the learning rate gradually decayed from 10^{-1} to 10^{-4}. The network is implemented using Tensorflow package. All the experiments are conducted on a PC with an Nvidia GTX 1080 GPU. Once such a DCNN is trained, it can be used to determine whether $\tilde{\mathbf{B}}_i$ will be modified by the enhancement filtering.

V. Proposed Robust Image Watermarking Scheme over Facebook Shared Images

In this section, we give the details of the embedding and extraction functions of our proposed image watermarking scheme. The cover image $\mathbf{I}$ is assumed to be in the pixel domain. In other words, if $\mathbf{I}$ is of other formats, e.g., JPEG, it will be decoded into pixels first. For color images, we only use the Y channel for the actual embedding. Therefore, the cover image $\mathbf{I}$ is considered as 8-bit grayscale image in the rest of this paper. As Facebook mainly manipulates the uploaded images over DCT domain, the proposed watermarking is naturally designed to be carried out in the DCT domain as well. Also, to avoid Facebook-side resizing, we enforce that both the height and the width of $\mathbf{I}$ are no larger than 2048 pixels. In the case that one of the dimensions exceeds 2048 pixels, we recommend to apply a user-side downsampling operation.

A. Embedding Algorithm

The cover image $\mathbf{I}$ is first partitioned into a series of non-overlapping blocks of size 8×8, which matches the block size used for JPEG compression on Facebook. Let $\mathbf{x}$ be a generic pixel block, and $\mathbf{X} = \{X_{i,j}\}$ be the corresponding DCT coefficient block. Then each DCT coefficient block $\mathbf{X}$ is quantized by using a quantization table $\mathbf{Q} = \{Q_{i,j}\}$ associated with $QF = 71$, generating a quantized DCT block $\mathbf{B} = \{B_{i,j}\}$, where

$$B_{i,j} = \text{round}\Big(\frac{X_{i,j}}{Q_{i,j}}\Big). \tag{6}$$

The choice of such a quantization table is to minimize the distortion caused by the JPEG compression on Facebook, where the same $QF = 71$ is used. The subsequent data embedding is conducted over the quantized DCT block $\mathbf{B}$. One of the major contributions of this work is a strategy of identifying the robust blocks that can perfectly survive

from the lossy operations on Facebook. In the remaining of this subsection, we first present the data embedding method for a particular block, followed by the details on the robust block selection. Finally, we address the issue on how to recursively embed the location map generated in the process of message embedding, which is of practical importance to achieve satisfactory robustness.

1) Block Based Embedding Function $\mathcal{E}$: We define a function $\mathcal{E}$ that takes the block $\mathbf{B}$ as the cover and $\mathbf{M}$ as the message to produce a marked block $\mathbf{B}'$, namely,

$$\mathbf{B}' = \mathcal{E}(\mathbf{B}, \mathbf{M}), \tag{7}$$

where the resulting block $\mathbf{B}'$ should be a robust block and hence will keep intact upon the manipulations of Facebook. Without loss of generality, we assume $\mathbf{M} = \{M_1, M_2, \cdots, M_K\}$ is a message bitstream of length K to be embedded. In fact, as long as $\mathbf{B}'$ is a robust block, it is not a challenging task for the data embedding. Though there are many ways to achieve this goal, in this work, we adopt the histogram shifting (HS) technique performed over the DCT domain for the actual data embedding, due to its excellent performance in the embedding capacity and preserving the image fidelity. Similar to many existing DCT-domain watermarking schemes, we embed the data in the AC coefficients, while leaving all the DC components unchanged. As a well-known fact, AC coefficients can be satisfactorily modelled with a Laplacian distribution [42], [43]. To avoid significant file size expansion, we also keep the zero AC coefficients (0 bin) unaltered, and only utilize the second highest peaks in the Laplacian distribution corresponding to $+1$ and -1 bins for the embedding purpose. Specifically, let $\mathbf{AC} = \{AC_1, AC_2, \cdots, AC_{63}\}$ be the quantized AC coefficients of the generic block $\mathbf{B}$. The data embedding is performed as follows:

$$AC_i' = \begin{cases} AC_i, & |AC_i| = 0 \\ AC_i + sign(AC_i) \times M_j, & |AC_i| = 1 \\ AC_i + sign(AC_i), & |AC_i| > 1 \end{cases} \tag{8}$$

where AC_i and AC_i' are the ith quantized AC coefficient before and after the embedding, respectively, and

$$sign(x) = \begin{cases} 1, & x \geq 0 \\ -1. & x < 0 \end{cases} \tag{9}$$

With such an embedding strategy, each value of the quantized AC coefficient AC_i changes at most 1, providing guaranteed high quality of the marked image regardless of the image content. This is of paramount importance for image sharing applications, where users seriously care the quality of their shared content. Furthermore, as will be shown later, we can ensure high-quality reconstruction of these quantized AC coefficients upon the data extraction, and in most of the cases, even perfect reconstruction.

Upon having the embedding function $\mathcal{E}$ for a particular block, we now present how to process all the blocks in the cover image to generate the marked image. Let $\mathbb{B} = \{\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_N\}$ be the quantized DCT blocks arranged in the raster scan order, and $\mathbf{M} = \{M_1, M_2, \cdots, M_K\}$ be the message bitstream to be embedded. The whole procedure is summarized into the following Algorithm 1, with graphic illustrations given in Fig. 6. More specifically, we first initialize a location map $\mathbf{L} = \{L_i\}_{i=1}^N = \mathbf{0}$, where $L_i = 0$ means no message embedded in the ith block. For each block $\mathbf{B}_i$, we then evaluate its potential embedding capacity c_i, which is the total number of $+1$ and -1 bins. In the rare case of $c_i = 0$, we simply let $\mathbf{B}_i' = \mathbf{B}_i$ without any embedding operations. If the capacity $c_i > 0$, we embed the message bits $\mathbf{M}_{(l_i+1)\rightarrow(l_i+c_i)}$ (the message segment indexed between $l_i + 1$ to $l_i + c_i$) into $\mathbf{B}_i$ and obtain the marked block $\mathbf{B}_i'$

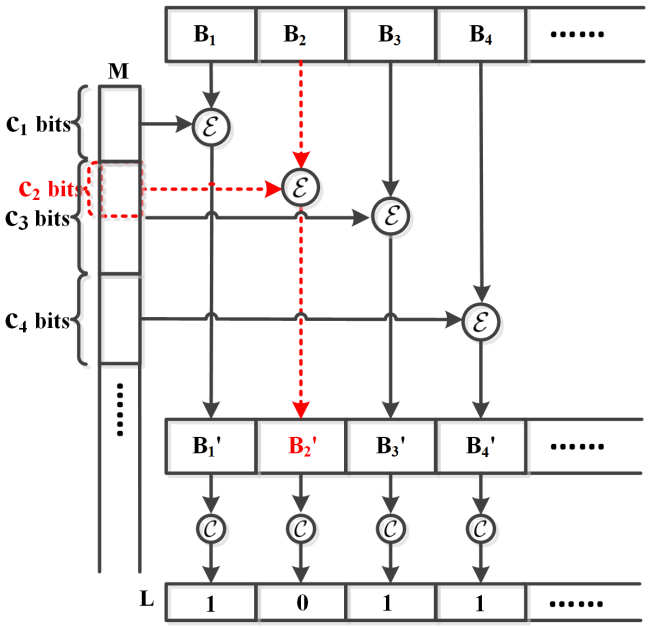


Fig. 6. Proposed image level embedding scheme.

$$\mathbf{B}_i' = \mathcal{E}(\mathbf{B}_i, \mathbf{M}_{(l_i+1)\rightarrow(l_i+c_i)}). \tag{10}$$

where l_i denotes the number of bits that have already been embedded previously, i.e.,

$$l_i = \sum_{n=1}^{i-1} c_n. \tag{11}$$

Meanwhile, we set $L_i = 1$ to indicate the existence of the embedding data. To achieve high robustness, the resulting $\mathbf{B}_i'$ needs to be further checked by the classifier designed in Section IV, and denote v_i as the classification result. If $v_i = 1$ (vulnerable block), then we give up the data embedding for the ith block and reverse the operation by setting $\mathbf{B}_i' = \mathbf{B}_i$, $c_i = 0$ and $L_i = 0$.

After the data embedding, an important side information that needs to be sent to the decoder side is the location map $\mathbf{L}$, which records whether a particular block contains message bits or not. In our scheme, we propose to use a recursive method to embed the location map into the cover image along with the message bits to achieve practically desirable robustness. The details will be given below.

2) Recursive Embedding of the Location Map: To embed the location map generated along with the data embedding, our strategy is to reserve a certain area of the cover image for it. To

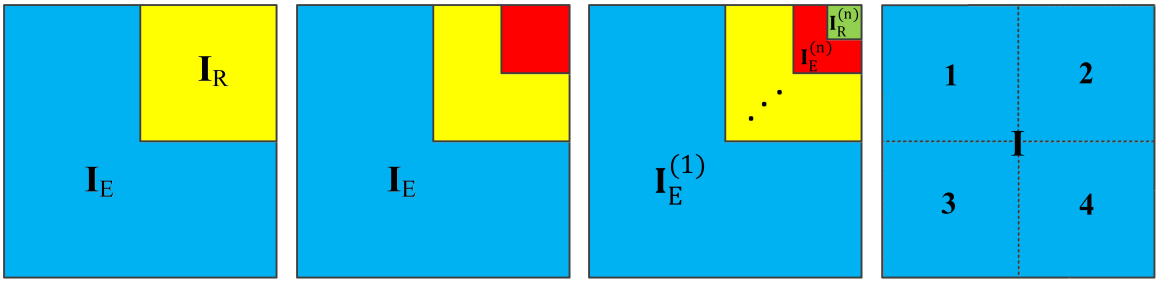


Fig. 7. Image partition of the embedding area and the reserved area.

Algorithm 1 Embed the message bitstream $\mathbf{M}$ into $\mathbb{B}$.

Input: The quantized DCT blocks $\mathbb{B} = \{\mathbf{B}_i\}_{i=1}^{N}$ and the binary message bitstream $\mathbf{M} = \{M_j\}_{j=1}^{K}$.

Output: Marked blocks $\mathbb{B}' = \{\mathbf{B'}_i\}_{i=1}^{N}$ and location map $\mathbf{L} = \{L_i\}_{i=1}^{N}$.

Initialize $\mathbf{L} = \mathbf{0}$ and the index for message bits $l = 0$.
for $i = 1$ to N **do**
  $c_i \leftarrow (\#+1 \text{ Bin in } \mathbf{B}_i) + (\#-1 \text{ Bin in } \mathbf{B}_i)$
  if $c_i > 0$ **then**
    $\mathbf{B'}_i \leftarrow \mathcal{E}(\mathbf{B}_i, \mathbf{M}_{(l+1)\rightharpoonup(l+c_i)})$.
    $v_i \leftarrow \mathcal{C}(\mathbf{B}'_i)$, where $\mathcal{C}(\cdot)$ is defined in (4).
    if $v_i = 0$ **then**
      $l \leftarrow l + c_i$.
      $L_i \leftarrow 1$.
    else
      $\mathbf{B}'_i \leftarrow \mathbf{B}_i$.
    end if
  else
    $\mathbf{B}'_i \leftarrow \mathbf{B}_i$.
  end if
end for
return $\{\mathbb{B}', \mathbf{L}\}$.

this end, the cover image $\mathbf{I}$ is first partitioned into two parts: the embedding area $\mathbf{I}_E$ and the reserved area $\mathbf{I}_R$, as shown in Fig. 7(a). The embedding area $\mathbf{I}_E$ is used to embed the message bits directly, and generates a location map $\mathbf{L}$, which will be compressed and then embedded into the reserved area $\mathbf{I}_R$. Clearly, due to the sparse nature of the location map, $\mathbf{I}_R$ should be much smaller than $\mathbf{I}_E$. Similarly, the embedding of the location map $\mathbf{L}$ in $\mathbf{I}_R$ generates a new location map, making it necessary to partition $\mathbf{I}_R$ again for the actual data embedding and the location map embedding, as can be seen in Fig. 7(b). Following this fashion, the reserved area needs to be recursively partitioned. Let $\mathbf{I}_E^{(i)}$ and $\mathbf{I}_R^{(i)}$ be the embedding and reserved areas in the ith iteration, respectively. The partition of $\mathbf{I}_R^{(i)}$ can be expressed as

$$\mathbf{I}_R^{(i)} = \mathbf{I}_E^{(i+1)} \bigcup \mathbf{I}_R^{(i+1)}. \tag{12}$$

The original cover image $\mathbf{I}$ can then be written into the following form:

$$\mathbf{I} = \Big(\bigcup_{i=1}^{n} \mathbf{I}_E^{(i)} \Big) \bigcup \mathbf{I}_R^{(n)}, \tag{13}$$

where n is the number of iterations. Such partition process can also be demonstrated in Fig. 7(c).

An important question remaining is how to perform the partition of the embedding and reserved areas. The following requirements should be fulfilled: 1) the reserved area should be large enough for embedding the location map generated in the same iteration, and 2) the partition should be perfectly reproducible at the decoder side. To be conservative for achieving the guaranteed robustness, the size of $\mathbf{I}_R^{(i)}$ is empirically set to one quarter of $\mathbf{I}_R^{(i-1)}$. In addition, ideally the reserved area should be located in the relatively high-capacity regions. This is because the message extraction is highly sensitive to the location map, and it should be guaranteed that we have enough room for embedding the location map. For determining the $\mathbf{I}_R^{(1)}$ in the first iteration, we partition the cover image $\mathbf{I}$ into four non-overlapping regions of the same size, denoted by $\mathbf{I}_1$, $\mathbf{I}_2$, $\mathbf{I}_3$, and $\mathbf{I}_4$, as shown in Fig. 7(d). In most cases, all of these four regions are sufficient to serve as the reserved area. However, in some extreme cases, if a large number of pixels in a region are close to 0 or 255, it will result in an exceptionally low capacity. This is because the blocks in such a region are more prone to overflow after performing quantization and embedding, thereby becoming vulnerable blocks and unable to carry any message bits. In order to avoid these low-capacity regions, $\mathbf{I}_R^{(1)}$ is selected to one of $\mathbf{I}_{\hat{k}}$ ($1 \le \hat{k} \le 4$), which is far away from the saturated pixel values, namely,

$$\hat{k} = \arg\min_{k} \sum_{i} \sum_{j} |\mathbf{I}_k(i,j) - 128|. \tag{14}$$

Once the index $\hat{k}$ is determined for the first iteration, it will be used for all the subsequent iterations. For instance, if the upper-right partition of $\mathbf{I}$ is selected to be $\mathbf{I}_R^{(1)}$, then in all the subsequent iterations, the upper-right one will be used as the reserved region.

Upon the image partition, we start the embedding operation for each area sequentially from $\mathbf{I}_E^{(1)}$. For ease of presentation, we here only consider full embedding, i.e., each embedding block carries the amount of bits at its capacity. After the embedding operation in $\mathbf{I}_E^{(1)}$, a location map $\mathbf{L}^{(1)}$ is generated, which will be compressed into $\tilde{\mathbf{L}}^{(1)}$ using an adaptive arithmetic coder `AOCoder` [44]. Then, $\tilde{\mathbf{L}}^{(1)}$ is embedded into $\mathbf{I}_E^{(2)}$. In general, a compressed location map $\tilde{\mathbf{L}}^{(i)}$, which is generated by $\mathbf{I}_E^{(i)}$, will be embedded in $\mathbf{I}_E^{(i+1)}$. As mentioned before and will be verified experimentally, the capacity of $\mathbf{I}_E^{(i+1)}$ is much

larger than the length of $\tilde{\mathbf{L}}^{(i)}$. Therefore, in order to maximize the embedding capacity, the excessive volume of $\mathbf{I}_E^{(i+1)}$ is used to embed the remaining bits in $\mathbf{M}$. Such process repeats until the size of $\mathbf{I}_R^{(n)}$ is below 32×32, and the resulting $\mathbf{I}_R^{(n)}$ is called the *seed* block. Before embedding data into the seed block $\mathbf{I}_R^{(n)}$, we first embed twelve consecutive 1's as a verification code to ensure the correct positioning at the decode side. The resulting compressed location map $\tilde{\mathbf{L}}^{(n)}$ is certainly shorter than 2 bytes and can be uploaded to Facebook as the metadata. Once all the embedding operations have been completed, the quantized DCT coefficients are applied a round of entropy coding which is exactly the same as the one used in the classic JPEG compression, to generate the final JPEG file $\mathbf{I}'$ to be shared over Facebook.

B. Data Extraction Algorithm

At the receiver side, Bob downloads the image $\bar{\mathbf{I}}$ in JPEG format, which is a lossy version of $\mathbf{I}'$ with hidden data, together with the metadata from Facebook. He attempts to recover the embedded message and restores the cover image simultaneously. Essentially, the data extraction and image reconstruction are the inverse of the embedding operations at Alice's side.

Let us first focus on a particular block $\bar{\mathbf{B}}$ of size 8×8, and discuss the data extraction and image restoration. We will then present how to process all the blocks of $\bar{\mathbf{I}}$. Obviously, we need to manipulate the quantized AC coefficients associated with the block, as only these coefficients carry the hidden data and are modified due to the embedding. Let $\bar{\mathbf{AC}} = \{\bar{AC}_1, \bar{AC}_2, \cdots, \bar{AC}_{63}\}$ be the quantized AC coefficients for $\bar{\mathbf{B}}$, which are entropy decoded from the downloaded JPEG file. Let also $\mathbf{M}' = \{M'_1, M'_2, \cdots, M'_K\}$ be the message bitstream to be extracted. From the HS operations described in (8), it is easy to know that the message bit carried by $\bar{AC}_i$ (denoted by M'_j) can be extracted as

$$M'_j = \begin{cases} 0, & |\bar{AC}_i| = 1 \\ 1. & |\bar{AC}_i| = 2 \end{cases} \tag{15}$$

Let $\hat{\mathbf{AC}} = \{\hat{AC}_1, \hat{AC}_2, \cdots, \hat{AC}_{63}\}$ be the AC coefficients to be restored. Upon the message bits extraction, the restoration can be conducted as follows:

$$\hat{AC}_i = \begin{cases} \bar{AC}_i, & |\bar{AC}_i| \le 1 \\ \bar{AC}_i - sign(\bar{AC}_i). & |\bar{AC}_i| > 1 \end{cases} \tag{16}$$

To process all the blocks, we first divide the image $\bar{\mathbf{I}}$ into four non-overlapping regions, in the exactly the same way as done in Fig. 7(d). The determination of the reserved region is conducted by resorting to (14). Note that the lossy operations of Facebook would lead to the distortion in $\bar{\mathbf{I}}$ with respect to $\mathbf{I}'$, and hence, the determination of the reserved region using (14) based on $\bar{\mathbf{I}}$ could be different from the one at the Alice's side. The verification code consisting of twelve consecutive 1's then plays an important role. If the determination using (14) is correct, the verification code must be encountered at the seed block $\bar{\mathbf{I}}_R^{(n)}$. Otherwise, the determination must be wrong, and re-searching needs to be done for the remaining three regions. Experimentally, we find that, in most cases, the determination using (14) is correct, and only in some rare cases, we need to perform the re-searching. The verification code, though seemly redundant, offers the practical guarantee for the robustness. Upon finding the embedding area and the reserved area, we can also confirm the seed block. In addition, from the metadata, we can apply the adaptive arithmetic decoder `A0Coder` to obtain the location map $\mathbf{L}'^{(n+1)}$ [44]. With $\mathbf{L}'^{(n+1)}$, the embedded bits can be easily extracted. Following the same procedure, we can readily recover all the message bits as well as the cover image $\hat{\mathbf{I}}$ recursively, through applying the de-quantization and IDCT.

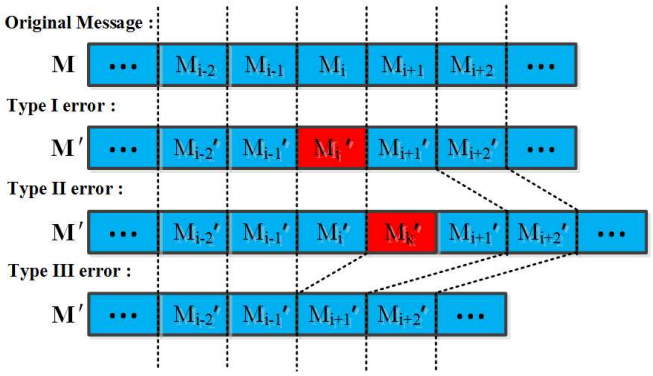


Fig. 8. Three types of data extraction errors in lossy channel.

C. Extraction Error Analysis

As mentioned earlier, the robustness of our proposed watermarking scheme over a practically lossy channel provided by Facebook does *not* rely on any ECCs. We now present some analyses on the potential message extraction errors, which can be categorized into the following types (see Fig. 8 for some illustrations).

Type I error: $|\mathbf{M}| = |\mathbf{M}'|$, but there exist some i's such that $M_i \neq M'_i$. This type of error is called flipping error.

Type II error: $|\mathbf{M}| < |\mathbf{M}'|$, meaning that the number of the extracted message bits is larger than the original one. This type of error is then called bit insertion error.

Type III error: $|\mathbf{M}| > |\mathbf{M}'|$, meaning that the number of the extracted message bits is smaller than the original one. This type of error is then called bit deletion error.

Both Type II and III errors lead to the de-synchronization problem, in which the extracted bits will be shifted forward or backward, though the message bits extraction is successful in many blocks. In addition, the Type I error could be a combination of Type II and III errors in multiple segments. Considering the image uploaded, the message is carried by some specific AC coefficient bins (bins $+1$, $+2$, -1, and -2). These bins can be grouped into the inner region and the rest into the outer region. In our proposed method, if a single error is assumed, the Type I error is caused by the distortions which modify an inner region coefficient to another inner region coefficient. Meanwhile, the type II error is caused by the distortions which modify an outer coefficient to an inner

coefficient, and the type III error is just the opposite of type II. The distortions which modify an outer coefficient to another outer coefficient will have no impact on the embedded data.

Once the de-synchronization occurs, the extraction accuracy could be very low, as the extraction correctness is usually defined by the bit-wise comparison. In addition, from the perspective of ECC, the de-synchronization errors are usually difficult to be corrected, especially when such types of errors could occur more than once. Though it is possible to design an ECC with the capability of correcting very limited number of delection/insersion errors [31], [45], it will significantly reduce the embedding capacity.

In our proposed method, however, we attempt to identify those robust blocks that do not change by the Facebook manipulations, based on our knowledge on these operations and the assistance of a DCNN-based powerful classifier. The performance of the DCNN-based classifier is critical in the overall robustness. The more vulnerable blocks are detected and skipped in the embedding phase, the higher robustness we can achieve. Since the functions $\mathcal{F}_F$ and $\mathcal{F}_J$ can be perfectly predicted at the client side, the vulnerable blocks caused by these two operations can be identified with 100% accuracy. The remaining part of the vulnerable blocks caused by the enhancement filtering only accounts for a small portion of all image blocks, usually only 1.2% on average. Relying on the DCNN in the classifier $\mathcal{C}$, this part of vulnerable blocks can also be detected very accurately (99.69% accuracy, with detailed experiments to be given in the next Section). Furthermore, luckily, due to the appropriate robust design, the number of robust blocks is quite large, making the potential embedding capacity very high. As will be seen soon, the average embedding capacity can reach around 23 K bits for a 512×512 grayscale image, while ensuring almost perfect recovery of both the message bits and the cover image. In comparison, the state-of-the-art competitors can easily fail (extraction accuracy drops to 50%) in many cases.

VI. Extensions to Other OSN Platforms

In this section, we explain how to extend our proposed technique to other OSN platforms. Instead of extending to a certain platform, we would rather give a general extension strategy which is suitable to all the other OSNs.

As the image manipulations on different OSN platforms are slightly different, we first need to probe into the target platform to know the details on these manipulations. According to our observations, the operations conducted by mainstream OSNs include format conversion, size transformation, compression, and enhancement filtering, which are very similar to what are performed on Facebook. Regarding a specific platform, mostly, not all the aforementioned operations will be performed, but rather only several of them. For instance, Twitter does not apply enhancement filtering on user uploaded images. Therefore, the methodology of investigating the image manipulations on Facebook could be easily extended to the other platforms. In addition, similarly, the parameters employed in each operation can be estimated through an off-line training procedure.

Upon having the knowledge on how the target OSN platform manipulates the uploaded images, we then need to adjust the parameters of our algorithm developed based on Facebook. Clearly, the first parameter we have to adjust is the image resolution threshold. In principle, the resolution of the cover image $\mathbf{I}$ should be enforced or pre-resized to be no larger than a certain threshold, such that no platform-side image geometric transformation will be applied. For instance, the shorter edge of the image uploaded to Wechat should not exceed 1080 pixels. It should be noted that some OSN platforms are even sensitive to the shape of the uploaded images. For example, Instagram prefers to crop the uploaded image into a square, which should also be taken into consideration. In addition, the client-side quantization table used to convert the cover image $\mathbf{I}$ into DCT domain should also be re-set according to the one used in the target platform. This is critical to minimize the distortion caused by platform-side JPEG compression.

Further, the structure of the block classifier $\mathcal{C}$ has to be re-designed in accordance with the possibly new pipeline in the target platform. Specifically, some modules in Fig. 4 could be added or removed, if needed. For those OSN platforms adopting the enhancement filtering, the deep model for enhancement filtering should also be adjusted by re-training with training samples from the target platform.

By adopting these aforementioned strategies, our proposed technique can be readily extended to other OSN platforms. Experimental results with two additional platforms: Wechat and Twitter, will be given in the next Section to validate this statement.

VII. Experimental Results

In this section, we experimentally evaluate the performance of our proposed watermarking scheme over Facebook shared images, and compare it with the state-of-the-art competing algorithms. The test set is composed of 200 images of sizes 512×512 with various characteristics. Some representative test images are demonstrated in Fig. 9. We are very careful when selecting the methods to be compared. As far as we know, [4] and [33] are the only two works which investigated the robust watermarking over OSN shared images. Hence, they should definitely be included for comparison. As our method is essentially a DCT-domain image watermarking scheme, we also compare with a classic baseline watermarking method [43], which is performed over the DCT-domain with the HS technique. Such comparison can demonstrate the value of the robust design, which is a key element of our proposed scheme. Further, we select a representative robust watermarking method [31] for comparison as well. For fairness, we assume that all these methods know the QF selection mechanism on Facebook, which is certainly an advantage for them. All the parameters in these methods are set to be the recommended values. All the marked images are uploaded to Facebook and then are downloaded for the data extraction and image reconstruction. This implies that all the data are from the interaction with the real Facebook, instead of under some simulated environments.

TABLE I
EMBEDDING PERFORMANCE COMPARISON (IN BITS) WITH [4], [33], [31], AND [43].

	Proposed		Ning [4]		Morkel [33]		Solanki [31]		Huang [43]	
Image	Capacity	Accuracy	Capacity	Accuracy	Capacity	Accuracy	Capacity	Accuracy	Capacity	Accuracy
Baboon	38832	100%	22037	50.29%	1060	100%	6356	55.19%	43502	52.59%
Portofino	22436	100%	7849	50.96%	1625	100%	2795	57.86%	23358	50.35%
Lena	19576	99.9949%	9377	50.65%	1476	100%	2305	58.16%	20359	53.15%
Goldhill	28176	100%	9855	50.04%	2020	100%	3342	53.60%	28776	68.57%
Programmer	20646	100%	8954	51.82%	1541	100%	3012	54.85%	22772	49.71%
Building	25168	100%	12859	50.87%	1425	100%	4197	52.13%	27035	56.58%
Husky	25647	100%	9796	50.41%	1243	100%	3179	52.81%	26615	50.23%
Cosmos	31051	99.9968%	15704	51.65%	1184	100%	5090	54.67%	34187	54.55%
Average	23099	99.9991%	12165	51.01%	1503	100%	3912	54.38%	27719	55.47%

(a) Baboon (b) Portofino (c) Lena (d) Goldhill

(e) Programmer (f) Building (g) Husky (h) Cosmos

Fig. 9. Some representative test images.

TABLE II
ACCURACY OF THE BLOCK CLASSIFIER $\mathcal{C}$.

	$y_i = 0$	$y_i = 1$
$y_i^* = 0$	93.57%	6.43%
$y_i^* = 1$	0.31%	99.69%

A. Evaluation on the performance of the DCNN-based classifier

We first evaluate the performance of our proposed DCNN-based classifier. The overall accuracy of the DCNN-based classifier is as high as 95.98%. In addition, the specific classification results are tabulated in Table II, where y_i^* and y_i are the true and the predicted labels, respectively. As we can see, the probability that a vulnerable block ($y_i^* = 1$) is miss-classified as a robust block ($y_i = 0$) is only 0.31%. This guarantees the high accuracy of the extracted message since 99.69% vulnerable blocks caused by the enhancement filtering are classified correctly. Meanwhile, 93.57% robust blocks are classified correctly and will be used to carry the message bits. This supports the high capacity of the proposed method.

B. Evaluation on the Embedding Capacity and Extraction Accuracy

We now investigate the embedding capacity and extraction accuracy of our proposed watermarking scheme. In Table I, we tabulate the embedding capacity and data extraction accuracy τ of the aforementioned methods, where τ is defined as

$$\tau = \frac{\#\text{ of correctly extracted bits}}{\#\text{ of embedded bits}}. \tag{17}$$

For each method, we present the results of the 8 representative test images illustrated in Fig. 9 and the average results over all 200 test images. As can be observed, our method can embed 23099 message bits on average for 512×512 images, while ensuring 100% accuracy of data extraction for most test images. The average extraction accuracy of our method is as high as 99.9991%. In contrast, the average extraction accuracy achieved by [43] and [4] is consistently much lower than 100%, even close to 50%, as they do not take any measures to deal with the distortions caused by Facebook. The embedding capacity of [43] is superior, which is a little bit higher than that of our method. Such high capacity is achieved by embedding message bits in all the DCT blocks indiscriminately without considering the robustness issues. As for the method [4], the data embedding is conducted by replacing the 2nd-LSB of all the quantized AC coefficients. Thus, the message bits can only be embedded in some large AC coefficients. Due to the distribution of the AC coefficients, naturally, the embedding capacity of [4] is much lower than that of our method. The accuracy of the method [33] is very attractive, i.e., 100%, meaning the perfect extraction. However, such results are obtained under some unrealistic assumptions, e.g., the location of the cover block and the original image are assumed to be available when performing the extraction. In addition, the capacity of the method [33] is rather low, only 1503 bits on average. Meanwhile, the robustness of the method [31] is achieved by a Repeat-Accumulate (RA) code. Since it is not particularly designed for OSNs, both the capacity and the accuracy are quite low.

Before ending the discussion on the embedding capacity and extraction accuracy, we also provide some experimental results on the proportion of the location map and the net capacity (gross capacity = net capacity + location map). In Section V, we state that the capacity of $\mathbf{I}_E^{(i+1)}$ is always larger than the length of $\tilde{\mathbf{L}}^{(i)}$. This is a necessary condition for our method to work properly. Fortunately, our approach satisfies this assumption very well. As illustrated in Table III, the average gross capacity of our method is 24938 bits while the average length of the location map of the whole image is only 1839 bits. The former is nearly 14 times larger than the latter.

(a) Original (b) Reference (QF=71) (c) Fully embedded (d) Restored

Fig. 10. Image visual quality comparison at different phases. The test image Lena is enlarged for better viewing experience.

TABLE III
SOME EXPERIMENTAL RESULTS ON GROSS CAPACITY AND NET CAPACITY (IN BITS).

Image	Gross Capacity	Net Capacity	Location Map
Baboon	40143	38832	1311
Portofino	23116	22436	680
Lena	20135	19576	559
Goldhill	28527	28176	351
Programmer	21801	20646	1155
Building	27266	25168	2098
Husky	27281	25647	1634
Cosmos	31950	31051	899
Average	24938	23099	1839
Ratio	100%	92.63%	7.37%

In our proposed embedding operation, the ratio of $\mathbf{I}_E^{(i-1)}$ to $\mathbf{I}_E^{(i)}$ is only 4:1 in each iteration. In addition to the relatively higher capacity of the reserved area, even in extreme cases, we can still guarantee that the capacity of $\mathbf{I}_E^{(i+1)}$ is much larger than the length of $\tilde{\mathbf{L}}^{(i)}$.

C. Evaluation on Image Quality

We further evaluate the image quality in different phases of our proposed scheme, where the original cover image $\mathbf{I}$ is used as the reference. First of all, we compare the quality of the Facebook shared image $\bar{\mathbf{I}}$ generated by our proposed method, with those produced by the other competing schemes. The quality of the JPEG coded image with $QF = 71$ is also shown as a reference. As can be seen from Fig. 11, for all the methods, the image quality decreases with the increasing embedding payload, and the PSNR values of the marked images are consistently lower than those of the JPEG coded images with $QF = 71$. This is reasonable, as any embedding can be regarded as a type of distortion. Compared with the methods in [4] and [31], the rate-distortion (RD) performance of our method is better, especially the performance improvement over [31] is quite significant. Meanwhile, as explained previously, our method also enjoys much higher embedding capacity as well as significantly improved extraction accuracy. When comparing with the method [43], its RD performance is better than ours at the low embedding rate regime; but the medium and high rate regimes become unachievable. In addition, the method [33] is superior than ours in terms of the RD performance. Nonetheless, the embedding capacity is much lowered. Finally, an example using the test image Lena is given in Fig. 10. As can be observed, the shared image is

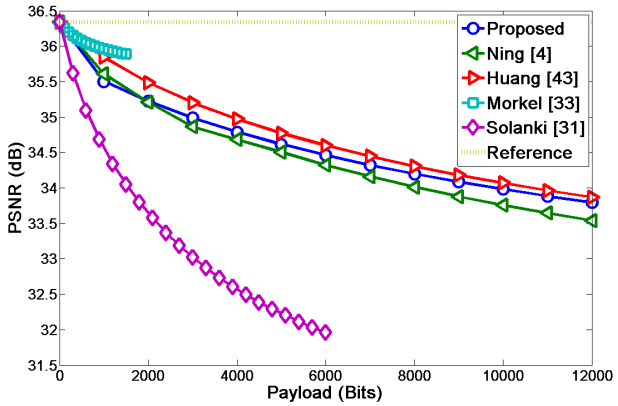


Fig. 11. RD curves for different methods.

of high quality and is visually similar to the JPEG coded one with $QF = 71$.

We now further evaluate the quality of the reconstructed image $\hat{\mathbf{I}}$. As shown in Table IV, the restored images of our proposed method are of high quality, in the sense that the PSNR degradations are less than 0.01 dB on average, compared with the reference JPEG coded images ($QF = 71$). In contrast, except for the method [43], the other three methods all cause severe PSNR drops. Especially, the PSNR degradation for the method [31] is as large as 4.38 dB on average, which is very significant. An example of the restored image lena is illustrated in Fig. 10 (d).

D. Evaluation on Storage Overhead and Running Time

We further investigate the storage overhead on Facebook incurred by different methods. The overhead is defined as the file size expansion of the shared image $\bar{\mathbf{I}}$ with respect to the corresponding offline JPEG image with $QF = 71$. Clearly, the overhead becomes larger as the payload increases, as shown in Fig. 12. Quantitatively, to embed 1 bit message, it will result in 1.2 bits storage overhead on average. Compare with [43], our proposed method exhibits almost the same performance. In contrast, the remaining three competing methods [4], [31], [33] outperform our method and almost incur no storage overhead. This is because all these methods adopt bits replacement strategy, and hence, do not lead to any file size expansion. For example, the method [4] replaced the 2nd-LSBs of the DCT coefficients with the message bits. The advantage regarding

TABLE IV
RESTORED IMAGE QUALITY (PSNR IN dB), COMPARED WITH [4], [33], [31], AND [43].

Image	JPEG(QF=71)	Proposed	Ning [4]	Morkel [33]	Solanki [31]	Huang [43]
Baboon	30.64	30.64	28.67	29.89	26.23	30.64
Portofino	35.51	35.51	33.09	34.77	31.05	35.51
Lena	37.40	37.40	34.46	36.64	32.82	37.40
Goldhill	35.27	35.27	32.98	34.83	30.51	35.27
Programmer	38.46	38.46	34.76	37.74	33.77	38.46
Building	37.72	37.72	34.16	37.04	34.04	36.58
Husky	41.50	41.50	36.82	40.85	36.65	41.50
Cosmos	36.46	36.46	33.75	35.67	31.82	36.46
Average	36.34	36.34	33.59	35.93	31.96	36.34

TABLE V
EXPERIMENTAL RESULTS ON COLOR IMAGES WITH MULTIPLE RESOLUTIONS OVER FACEBOOK, WECHAT AND TWITTER.

	Facebook			Wechat			Twitter		
Resolution	Capacity	Accuracy	Quality($\hat{\mathbf{I}}$)	Capacity	Accuracy	Quality($\hat{\mathbf{I}}$)	Capacity	Accuracy	Quality($\hat{\mathbf{I}}$)
360×480	10776	99.9994%	33.23	10993	100%	34.35	13835	100%	35.65
480×640	18508	100%	34.18	19144	100%	35.40	23938	100%	36.49
640×720	26682	100%	35.07	27663	100%	36.18	34644	100%	37.25
720×960	37335	99.9988%	35.89	40337	100%	36.97	50499	100%	38.32
960×1080	50328	99.9995%	36.84	58274	100%	37.72	73681	100%	39.06
1080×1920	96178	99.9992%	38.11	109660	100%	39.15	140297	100%	40.59

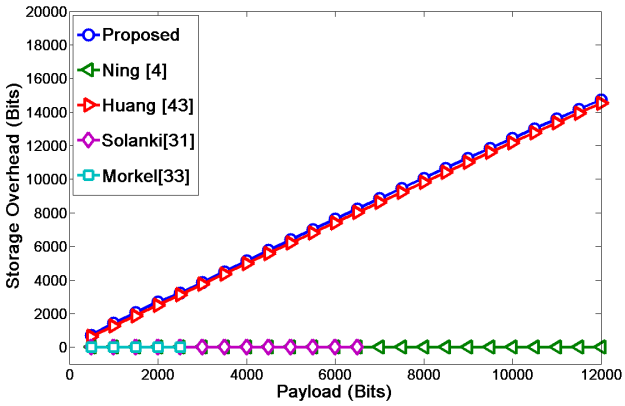


Fig. 12. Rate-Overhead curves for different methods.

TABLE VI
RUNNING TIME COMPARISON WITH [4], [33], [31], AND [43].

Method	Main Components	Ave. Time
Proposed	DCT+Embedding+Block Classifier	1.12 s
Ning [4]	DCT+Embedding	0.82 s
Solanki [31]	DCT+ECC+AC Selection+Embedding	1.29 s
Morkel [33]	Block Selection+Embedding	0.26 s
Huang [43]	DCT+Block Selection+Embedding	1.19 s

(a) (b) (c) (d)

Fig. 13. Some representative color test images.

the storage overhead is achieved at very low message bits extraction accuracy, as discussed in Section VI-B.

We also report the running time for the compared algorithms. The results are illustrated in Table VI, where the experiments are conducted on Matlab 2013b and Python 3.6 in a personal PC with Intel i7@3.40 GHz CPU and 8 GB RAM. All the presented numbers are obtained from averaging over the 200 images in the test set. It takes 1.12 seconds on average for our method to complete the full embedding, while the methods [43] and [31] take 1.29 seconds and 1.19 seconds, respectively. In contrast, the approach [4] is slightly more efficient due to the simpler embedding strategy. Among all the compared algorithms, the one in [33] is the fastest, taking 0.26 seconds only. This is because the embedding is carried out in the pixel domain via simple operations.

E. Evaluation on Extensibility

To evaluate the extensibility of our proposed scheme to other OSN platforms, color images, and images with different resolutions, we build a new dataset formed by 600 color images with various resolutions ranging from 360×480 to 1080×1920. Some representative color test images are shown in Fig. 13. We also demonstrate the extensibility by considering two extra platforms, i.e., Wechat and Twitter. The experimental results on embedding capacity, extraction accuracy and restored image quality (PSNR in dB), are illustrated in Table V. As can be seen, Twitter seems to the most embedding-friendly platform, with the highest embedding capacity, 100% extraction accuracy, and the best reconstructed image quality. The performance on Facebook and Wechat is similar. This indicates that our proposed technique works quite well on color images over different OSN platforms. Meanwhile, the embedding capacity scales up satisfactorily with the increasing spatial resolution.

VIII. CONCLUSIONS

In this paper, we have designed a robust and high-capacity image watermarking scheme over Facebook shared images.

To combat with the various lossy operations conducted by Facebook, we have first recovered the image manipulation mechanism through a DCNN approach. Based on the knowledge on the lossy channel offered by Facebook, we have designed a DCT-domain image watermarking scheme, which is rather robust even without the help of an ECC. We also have extended our proposed scheme to various OSN platforms, such as Twitter and Wechat. We have conducted extensive experiments to validate the superior performance of our proposed method in terms of the embedding capacity, data extraction accuracy, and the quality of the reconstructed images.

REFERENCES

[1] Facebook, Newsroom[Online]. Available: http://newsroom.fb.com/company-info/, accessed Mar. 2017.

[2] Fast Technology, [Online]. Available: http://news.mydrivers.com/1/483/483952.htm, accessed Mar. 2017.

[3] M.-R. Ra, R. Govindan, and A. Ortega, "P3: Toward privacy-preserving photo sharing." in *Proc. 10th USENIX Symp. Netw. Syst. Design Implem. (NSDI)*, 2013, pp. 515–528.

[4] J. Ning, I. Singh, H. V. Madhyastha, S. V. Krishnamurthy, G. Cao, and P. Mohapatra, "Secret message sharing using online social media," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, 2014, pp. 319–327.

[5] W. Sun, J. Zhou, R. Lyu, and S. Zhu, "Processing-aware privacy-preserving photo sharing over online social networks," in *Proc. ACM Int. Conf. Multimedia (ACMMM)*, 2016, pp. 581–585.

[6] M. Urvoy, D. Goudia, and F. Autrusseau, "Perceptual dft watermarking with improved detection and robustness to geometrical distortions," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 7, pp. 1108–1119, 2014.

[7] X. Kang, J. Huang, and W. Zeng, "Efficient general print-scanning resilient data hiding based on uniform log-polar mapping," *IEEE Trans. Inf. Forensics Secur.*, vol. 5, no. 1, pp. 1–12, 2010.

[8] M. Zareian and H. R. Tohidypour, "A novel gain invariant quantization-based watermarking approach," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 11, pp. 1804–1813, 2014.

[9] J. Cao and J. Huang, "Controllable secure watermarking technique for tradeoff between robustness and security," *IEEE Trans. Inf. Forensics Secur.*, vol. 7, no. 2, pp. 821–826, 2012.

[10] X. Wang, J. Wu, and P. Niu, "A new digital image watermarking algorithm resilient to desynchronization attacks," *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 4, pp. 655–663, 2007.

[11] H. Fang, W. Zhang, H. Zhou, H. Cui, and N. Yu, "Screen-shooting resilient watermarking," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 6, pp. 1403–1418, 2018.

[12] C.-Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y. M. Lui, "Rotation, scale, and translation resilient watermarking for images," *IEEE Trans. Image Process.*, vol. 10, no. 5, pp. 767–782, 2001.

[13] P. Dong, J. G. Brankov, N. P. Galatsanos, Y. Yang, and F. Davoine, "Digital watermarking robust to geometric distortions," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2140–2150, 2005.

[14] J. S. Seo and C. D. Yoo, "Image watermarking based on invariant regions of scale-space representation," *IEEE Trans. Signal Process.*, vol. 54, no. 4, pp. 1537–1549, 2006.

[15] C.-W. Tang and H.-M. Hang, "A feature-based robust digital image watermarking scheme," *IEEE Trans. Signal Process.*, vol. 51, no. 4, pp. 950–959, 2003.

[16] H. S. Kim and H.-K. Lee, "Invariant image watermark using zernike moments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 766–775, 2003.

[17] Y. Xin, S. Liao, and M. Pawlak, "A multibit geometrically robust image watermark based on zernike moments," in *Proc. 17th IEEE Int. Conf. Pattern Recogn. (ICPR)*, vol. 4, 2004, pp. 861–864.

[18] M. Alghoniemy and A. H. Tewfik, "Geometric invariance in image watermarking," *IEEE Trans. Image Process.*, vol. 13, no. 2, pp. 145–153, 2004.

[19] V. Jayanthi, V. Selvalakshmi, and V. Rajamani, "Digital watermarking robust to geometric distortions in biomedical images," in *Proc. IEEE Int. Conf. Contr. Autom. Commun. Energy Conservation (INCACEC)*, 2009, pp. 1–6.

[20] L. Li, S. Li, A. Abraham, and J.-S. Pan, "Geometrically invariant image watermarking using polar harmonic transforms," *Inf. Sci.*, vol. 199, pp. 1–19, 2012.

[21] S. Golabi, M. S. Helfroush, H. Danyali, and M. Owjimehr, "Robust watermarking against geometric attacks using partial calculation of radial moments and interval phase modulation," *Inf. Sci.*, vol. 269, pp. 94–105, 2014.

[22] S. Golabi, M. S. Helfroush, and H. Danyali, "Non-unit mapped radial moments platform for robust, geometric invariant image watermarking and reversible data hiding," *Inf. Sci.*, vol. 447, pp. 104–116, 2018.

[23] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Q. Sun, and X. Lin, "Robust lossless image data hiding designed for semi-fragile image authentication," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 4, pp. 497–509, 2008.

[24] M. D. Swanson, B. Zhu, and A. H. Tewfik, "Robust data hiding for images," in *Proc. IEEE Digital Signal Process. Workshop (DSPWS)*, 1996, pp. 37–40.

[25] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Q. Sun, and X. Lin, "Robust lossless image data hiding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, vol. 3, 2004, pp. 2199–2202.

[26] X.-T. Zeng, L.-D. Ping, and X.-Z. Pan, "A lossless robust data hiding scheme," *Pattern Recogn.*, vol. 43, no. 4, pp. 1656–1667, 2010.

[27] N. Abdulaziz and K. K. Pang, "Robust data hiding for images," in *Proc. IEEE Int. Conf. Commun. Technol. (ICCT)*, vol. 1, 2000, pp. 380–383.

[28] A. Aung, B. P. Ng, and S. Rahardja, "A robust watermarking scheme using sequency-ordered complex hadamard transform," *J. Signal Process. Syst.*, vol. 64, no. 3, pp. 319–333, 2011.

[29] R. Franklin, G. Manekandan, and V. Santhi, "Entropy based robust watermarking scheme using hadamard transformation technique," *Int. J. Computer Appl.*, vol. 975, p. 8887, 2011.

[30] E. Etemad, S. Samavi, S. R. Soroushmehr, N. Karimi, M. Etemad, S. Shirani, and K. Najarian, "Robust image watermarking scheme using bit-plane of hadamard coefficients," *Multimedia Tools Appl.*, vol. 77, no. 2, pp. 2033–2055, 2018.

[31] K. Solanki, N. Jacobsen, U. Madhow, B. Manjunath, and S. Chandrasekaran, "Robust image-adaptive data hiding using erasure and error correction," *IEEE Trans. Image Process.*, vol. 13, no. 12, pp. 1627–1639, 2004.

[32] B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1423–1443, 2001.

[33] T. Morkel, "The osn-tagging scheme: Recoverable steganography for online social networks," in *Proc. IEEE Int. Conf. Next Generation Comput. Appl. (NextComp)*, 2017, pp. 11–16.

[34] J. Tao, S. Li, X. Zhang, and Z. Wang, "Towards robust image steganography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 2, pp. 594–600, 2018.

[35] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP J. Inf. Secur.*, vol. 2014, no. 1, pp. 1–13, 2014.

[36] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, "Using statistical image model for jpeg steganography: uniform embedding revisited," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 12, pp. 2669–2680, 2015.

[37] W. Sun, J. Zhou, S. Zhu, and Y. Y. Tang, "Robust privacy-preserving image sharing over online social networks (osns)," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 14, no. 1, pp. 1–14, 2018.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Sys. (NIPS)*, 2012, pp. 1097–1105.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learning (ICML)*, 2015, pp. 448–456.

[40] G. Xu, H. Wu, and Y. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Process. Lett.*, vol. 23, no. 5, pp. 708–712, May 2016.

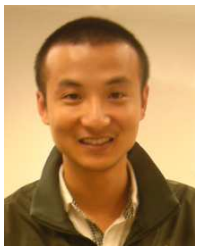
[41] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int. Conf. Comp. Vis. (ICCV)*, 2015, pp. 1026–1034.

[42] R. Reininger and J. Gibson, "Distributions of the two-dimensional dct coefficients for images," *IEEE Trans. Commun.*, vol. 31, no. 6, pp. 835–839, 1983.

[43] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in jpeg images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 9, pp. 1610–1621, 2016.

[44] Adaptive Arithmetic Coder, [Online]. Available: http://www.algorithmic-solutions.info/leda manual/A0Coder.html, 2017.

[45] P. J. Lee, "Error correction code for correcting shift and additive errors," Dec. 13 2011, uS Patent 8,078,943.



Weiwei Sun (S'14) received the B.S. degree from the College of Electronics and Information Engineering, Shenyang University, Shenyang, China, in 2012, the M.S. degree from the Department of Computer and Information Science, University of Macau, Macau, China, in 2015. He is currently working toward the Ph.D. degree with the Department of Computer and Information Science, and also with the State Key Laboratory of Internet of Things for Smart City, University of Macau, Macau, China. His research interests include the multimedia signal processing, online social networks, multimedia security and forensics.

Jiantao Zhou (M'11-SM'19) received the B.Eng. degree from the Department of Electronic Engineering, Dalian University of Technology, in 2002, the M.Phil. degree from the Department of Radio Engineering, Southeast University, in 2005, and the Ph.D. degree from the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, in 2009. He held various research positions with University of Illinois at Urbana-Champaign, Hong Kong University of Science and Technology, and McMaster University. He is an Associate Professor with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, and also the Interim Head of the newly established Centre for Artificial Intelligence and Robotics. His research interests include multimedia security and forensics, multimedia signal processing, artificial intelligence and big data. He holds four granted U.S. patents and two granted Chinese patents. He has co-authored two papers that received the Best Paper Award at the IEEE Pacific-Rim Conference on Multimedia in 2007 and the Best Student Paper Award at the IEEE International Conference on Multimedia and Expo in 2016. He is an Associate Editor of the IEEE TRANSACTIONS on IMAGE PROCESSING.

Yuanman Li (M'20) received the B.S. in software engineering from Chongqing University, Chongqing, China, in 2012, and the Ph.D. degree in computer science from University of Macau, Macau, China, in 2018. He joined Shenzhen University as a Assistant Professor in 2019. He is currently with College of Electronics and Information Engineering, Shenzhen University, Guangdong, China. His research interests include multimedia security, pattern recognition and machine learning.

Ming Cheung received his PhD in 2018 from HKUST. He is the top 3 of the 2018 PhD Research Excellence Award by HKUST School of Engineering. He successfully publishes several journal papers in top IEEE and ACM journals in the areas of Multimedia Big Data and Social Computing, such as IEEE TMM, IEEE Trans. Big Data and ACM ToMM. He is also the founder several startups that are granted with funding and investment from Hong Kong Science Park, Cyberport and investors.

James She is the founding director of HKUST Social Media Lab., and also associated with the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST). His current research interests include Social Computing and Multimedia, Data Science and AI for Art, IoT for Smart and Multimedia Systems. He is currently the associate editors for both IEEE Transactions on Multimedia and ACM Transactions on Multimedia Computing, Communications and Applications.