Reversible Data Hiding over Facebook

Ran Lyu[*], Jian-Tao Zhou, Wei-Wei Sun, Li Dong

Department of Computer and Information Science, University of Macau, Macau, China.

Abstract

Facebook, the most popular Online Social Network (OSN), could be used as a platform to share secret messages through JPEG images online. However, due to the various lossy operations conducted over Facebook, the data embedded into the JPEG images can be easily destroyed, making the data extraction infeasible. More importantly, all these operations are carried out without users' interference. In this paper, we first perform an in-depth investigation of the various lossy operations that Facebook applies to uploaded images. Based upon such prior knowledge, we propose a DCT-domain data hiding scheme that can effectively embed a large amount of data and successfully extract them out from the downloaded images, defeating the uncontrolled lossy operations. Compared with the state-of-the-art techniques, the proposed method offers much higher embedding capacity, and can extract the data successfully with very high probability. Furthermore, the restored image upon data extraction is of high quality, and the file size expansion is negligible. Extensive experimental results are provided to validate our findings.

Keywords: data hiding, JPEG, reversible data hiding, Facebook

1. Introduction

Facebook, the most popular Online Social Network (OSN), could be used as a platform to share secret messages through JPEG images online. The operations conducted by Facebook (including resizing, JPEG compression, etc) are much more complex than other social networks [1]. Although the idea of sharing the hidden message over Facebook is attractive, it faces many inevitable challenges. Huang [2] proposed a very robust steganography for pure JPEG. However, Huang's method is not robust to overflowing caused by re-compression on Facebook. Hiney [3] tested nine steganography methods on Facebook. The experimental results have shown that these methods work well offline, but perform poorly on Facebook. Castiglione analyzed how some prevalent OSNs processed the uploaded images [1] and proposed two steganography schemes [4] with low embedding capacity on Facebook. Nagaraja [5] proposed Stegobot, using the YASS [6] specifically, which has the limitation of the embedding capacity. Ning [7] proposed an approach 2-LSB inspired by LSB [8]. The drawbacks are two-fold: lower embedding capacity compared to LSB; lacks of discussion about visual quality. Amsden [9] proposed to transmit hidden information using JPHide and JPSeek [10] over a cover photo on Facebook (failed over album images).

In this work, we first perform an in-depth investigation of the various lossy operations that Facebook applies to uploaded images. Based upon such prior knowledge, we propose a DCT-domain data hiding scheme. Compared with the state-of-the-art techniques, the proposed method offers much higher embedding capacity, and can extract the data successfully with very high probability. The rest of the paper is organized as follows. The investigation of Facebook is given in Section 2. The proposed scheme and system are described in Section 3. Experimental results are shown in Section 4. Finally, the conclusion is given in the Section 5.

---

* Corresponding author. E-mail address: mb45442@umac.mo

2. Analysis of the Operations Conducted over Facebook

We upload 1338 gray uncompressed images from UCID-v2 [11] to Facebook, and then download them. The published images are compressed using quality factor (QF) chosen by Facebook. The quantization tables are extracted from the downloaded images. It is observed that all the quantization tables match the International JPEG Group standard [12]. It turns out that the employed QF varies from 71 to 92. We then make a hypothesis for the re-compressed image of Facebook: the QF chosen by Facebook should be smaller than or equal to the one in uploaded image. To verify this assumption, we compress images with QFs vary from 50 to 100 and upload them to Facebook. Then we download them and extract the quantization tables from the downloaded images. The statistical results agree with our assumption. We have an important observation: for JPEG images compressed by $QF \leq 71$, the QFs of downloaded image are all consistent with 71.

3. Facebook Message Sharing System (FMSS)

To clearly describe the Facebook Message Sharing System (FMSS), the related terms are defined: cover image, the original image; stego image, the embedded image; FBstego image, the published stego image over Facebook; restored image, the image restored from the FBstego image after extraction; QF_{embed}, the QF used in stego image offline; QF_{FB}, the QF used in the FBstego image on Facebook. The following roles are involved in the FMSS: sender, Facebook and receiver. Sender compresses the cover image with $QF_{embed}=71$ before embedding which forces Facebook chooses $QF_{embed}=QF_{FB}$ in re-compression. In FMSS, we fully trust the sender and receiver that they can act exactly as we wish. Facebook might damage the data hiding scheme accidentally. However, Facebook will follow a fixed process during re-compression. Therefore, we take the Facebook as semi-trusted.

3.1. Block selection algorithm

Even we make $QF_{embed}=QF_{FB}$, the Inverse and Forward Discrete Cosine Transform (IDCT and FDCT) in re-compression of Facebook still has a chance to cause the failure of data hiding. Given I' as the embedded image before the pixel operations in JPEG. The pixel operations include truncation and rounding, which cause the overflowing *Err: I'*-Round(Truncate(*I'*)). Since overflowing phenomenon is inevitable in JPEG algorithm, we check the embedded block whether it's overflow with large *Err* value or not. Not all blocks in an image have large *Err* of overflowing phenomenon and we call them stable blocks referring stable QF. The new selection method of stable blocks needs a map for each image to identity positions. It's need to reduce the additional storage introduced by map. We call embeddable blocks Qualified Block (QB) and others Unqualified Block (UB). A flag bit $b_F \in \{1,0\}$ needs to be inserted to determine the QBs and UBs. In this solution, the burden is only related to the number of embedded blocks, but not the size of the image. To achieve higher capacity in the embedding process, we adopt the same strategy proposed by Huang [2]: embedding bits in block with more zero coefficients.

3.2. Embedding algorithm

By spanning QB in zigzag, we build a qualified embedding coefficients vector Q and shifting coefficients vector S with size of n and m, respectively. Each secret message bit can be represented as $b \in \{1,0\}$. The proposed embedding algorithm for QB is as Eq. (1)-(3):

$$Q_i'=Q_i+sign(Q_i)\,b \quad i=1,...,n-1 \quad Q_i \in \{ \text{AC coefficients of each QB and } |Q_i|=1\} \tag{1}$$

$$S_j'=S_j+sign(S_j) \quad j=1,...,m \quad S_j \in \{ \text{AC coefficients of each QB and } |S_j|=2\} \tag{2}$$

$$Q_i'=Q_i+sign(Q_i)\,b_F \quad i=n \quad Q_i \in \{ \text{AC coefficients of each QB and } |Q_i|=1\} \tag{3}$$

The flag bit $b_F=1$ is embedded into the last Q_i by Eq. (3). To make sure of efficiency, there should be at least one valid message bit and the flag bit in a QB, so the length of Q should be greater than 1, *i.e.*, $n \leq 2$. The blocks with low capacity and overflowing phenomenon can be set as UBs. By spanning each UB in zigzag sequence, the first coefficient with the value of ± 1 or ± 2 counting backwards is u. For UB, we only need to embed the flag bit 0 by: $u'=sign(u)$. If there's no u in current UB, we can just do nothing and move to next block. Sender scans the blocks to check they're QBs or UBs by proposing block selection. Then message bits and flag bits are embedded following the embedding algorithm above. Not all blocks need to be scanned while embedding since the embedding process might finish before scanning all blocks. A reasonable choice for visual quality is to stop embedding scanning and keep the following blocks unchanged.

3.3. Data extraction and image restoration algorithm

Given the secret message M with the length of L. L can be represented by l bits. Here l is a fixed and predefined number. Therefore, sender embeds L and M with $l+L$ bits totally. In the experiment, we set $l=16$. The receiver downloads the FBstego image from Facebook and extracts the secret message bits from it. The receiver scans the blocks to check if they're QBs or UBs by extracting the flag bits. The receiver can calculate the length of the embedding bits L by first l extracted bits. Specifically, in message extraction part, the vector Q_i^* is built in each block and the restored AC coefficients $Q_k^{*\prime}$ can be calculated by Eq. (4).

$$Q_k^{*\prime}=sign(Q_k^*) \quad k=1,...,p \quad Q_k^* \in \{ AC\ coefficients\ that\ |Q_k^*|=1\ or\ |Q_k^*|=2 \} \tag{4}$$

If $p = 0$ or $Q_p^* = 1$ (it indicates $b_F=0$), the block is UB and we move to next block. Otherwise, take the block as QB. Each extracted bit b' in QBs is calculated by: $b'=0$ if $|Q_k^*|=1$ or $b'=1$ if $|Q_k^*|=2$. For each QB, the restored AC coefficients $Q_k^{*\prime}$ can be calculated by Eq. (5):

$$Qk^{*\prime}=sign(Qk^*) \tag{5}$$

The scanning procedure continues. Note that after the extraction of $l + L$ bits, the rest blocks won't be scanned and restored because there's no change in them during embedding process.

4. Experimental results

We select test images from UCID [11] and convert these color images into gray. The size of images in experiments is either $384 \cdot 512$ or $512 \cdot 384$. In our experiments, Facebook does not resize those images. We implement four competing methods: Huang's method [2]; LSB, 2-LSB and LSB+2-LSB based on [7]. Note that LSB+2-LSB are called MixLSB in our experiments. For fair comparison, these four competing methods (Huang's method [2], LSB[7], 2LSB[7] and MixLSB[7]) and our proposed method are all compressed with $QF_{embed}=71$ before embedding. The embedding bits are randomly generated and all cover images are compressed by JPEG standard [12].

4.1. Data extraction accuracy

The 1338 images in UCID [11] are all used for comparing accuracy. We use bit error rate (BER) to describe the performance of accuracy for each method, which is defined in Eq. (6).

$$BER=\#(error)/\#(embedded) \tag{6}$$

where $\#(\cdot)$ is the function of calculating the number of bits. Clearly, lower BER indicates better performance. As we can see from the Table 1, even excluding the influence of QF_{embed}, our method is still much better than other methods in terms of BER. The four compared methods are totally invalid when it comes to Facebook. Note that the random guess for each bit is about 0.5 and

four compared methods are all very close to this rate. Huang's method is designed for pure JPEG and not robustness to overflowing. Ning's methods are failed on the current Facebook system. One of potential reasons might be the updating of Facebook. The superior performance of our method is attributed to the proposed block selection algorithm. The selected blocks are robustness to re-compression of JPEG and other unknown functions on Facebook.

Table 1 Comparison of Bit Error Rate (BER) of 1338 images in UCID for each method

Method	Proposed	Huang[2]	LSB[7]	2LSB[7]	MixLSB[7]
BER	**0.18%**	44%	46%	48%	46%

4.2. Embedding capacity

Since the successfully embedded bits are almost zero for four competing methods, it has no need to discuss their embedding capacity performance. The 1338 images in UCID [11] are all used to evaluate the capacity of the proposed scheme. The average embedding capacity for our method is about 12064 bits in $384 \cdot 512$ image, which achieves a high embedding rate: 6.1%. The capacity of our method is related to the number Qualified Blocks (QBs), which has a lot to do with the content of the image itself. In other words, an image with more QBs is likely to achieve higher capacity. Therefore, sorting out high-capacity image based on the number of QBs can improve the average embedding capacity. We sort 1338 cover images from largest capacity to lowest and embed bits in high-capacity image. The refined average capacity is about 20000 bits in 200 images.

4.3. File size expansion

In order to evaluate the performance of file size expansion, we randomly select 100 images from UCID [11]. We only choose the size values of FBstego images in each method. We compress cover images with $QF_{embed}=71$ in JPEG standard and call it as JPEG71. We upload JPEG71 on Facebook and download to obtain FBJPEG71. We add JPEG71 images and FBJPEG71 images in comparison, because the ideal size is not only small, but also close to the same image without embedding rates. The variation of file size for different payloads is measured by bytes and the results are shown in Fig. 1. As can be seen, LSB, 2LSB and MixLSB have no increment in file size and are almost the same size. This is because these three methods use VLI coding [13] which won't affect the entropy codes of JPEG. JPEG71 and FBJPEG71 are unchanged for no embedding bits. Compared to Huang, we have smaller average file size and slower trend of growth. And most of file size values in proposed method are located in the space between FBJPEG71 to JPEG71, which contributes to covertness.
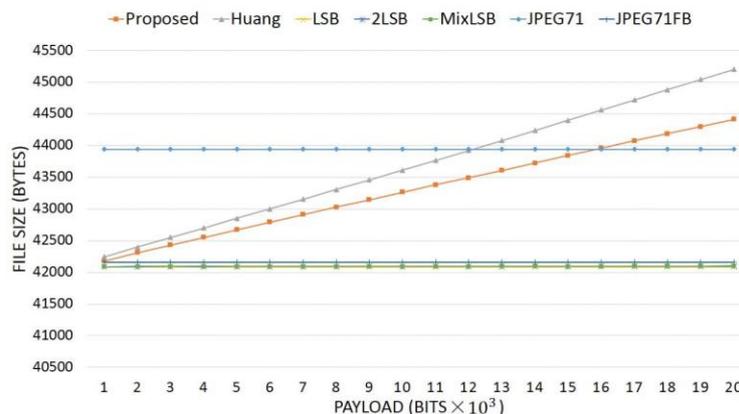

Fig. 1 File size (bytes) of FBstego image

4.4. Image visual quality

To evaluate the performance of visual quality, we randomly select 100 images from UCID [11]. The performance of visual quality is evaluated by Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). One example of the proposed method is shown in Fig. 2. These three types of images are also used in the four competing methods. The payloads in stego vary

from 1000 bits to 20000 bits. We use Mutual Information (*MI*) to describe the mutual dependence between the two random variables: *X* and *Y*. *X* is the real embedded sequence and *Y* is the sequence extracted by steganography. The mutual information of *X* and *Y* is donated by *I(X;Y)* and *H(·)* is the function of calculating the entropy of a random sequence. *MI* can be calculated by Eq. (7):

$$MI = I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{7}$$

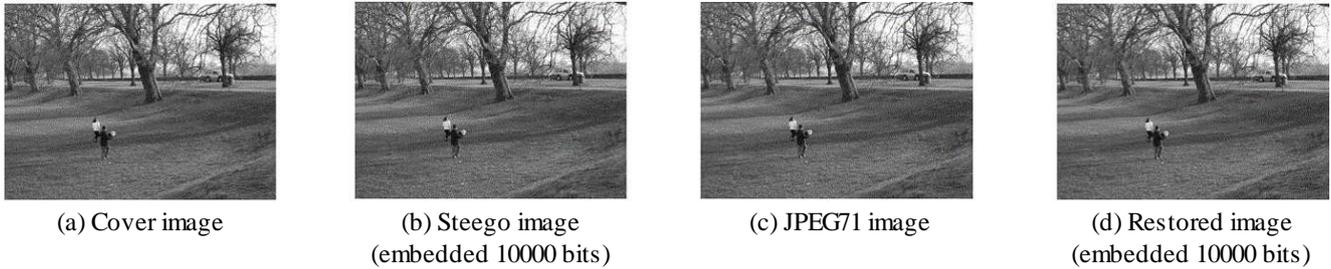| (a) Cover image | (b) Steego image (embedded 10000 bits) | (c) JPEG71 image | (d) Restored image (embedded 10000 bits) |

Fig. 2 One example of proposed method (a) Original image in steganography (b) The image embedded with 10000 bits by proposed method (c) The image restored form FBstego (d)The image compressed by JPEG in quality factor 71
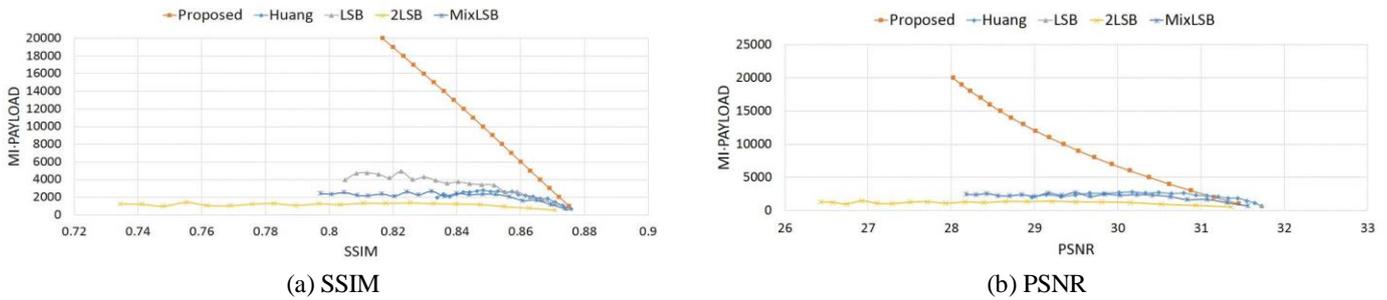

| (a) SSIM | (b) PSNR |

Fig. 4 Visual quality of stego images VS cover images

| (a) SSIM | (b) PSNR |
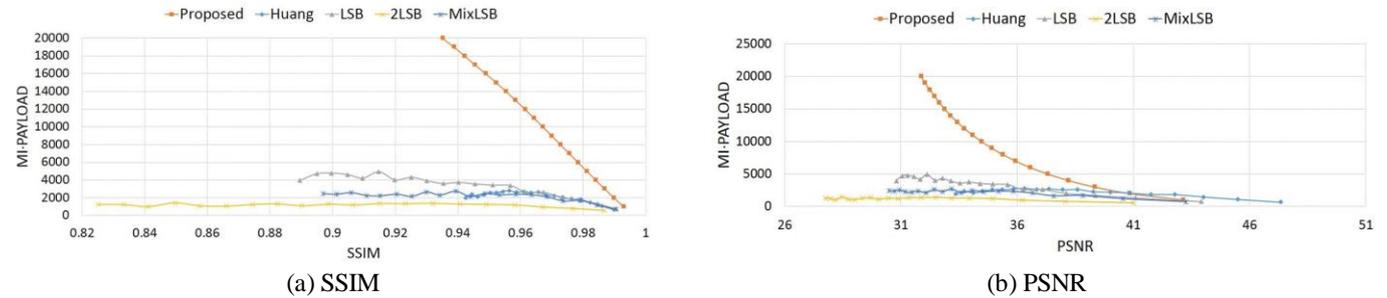
Fig. 5 Visual quality of stego images VS JPEG71 images
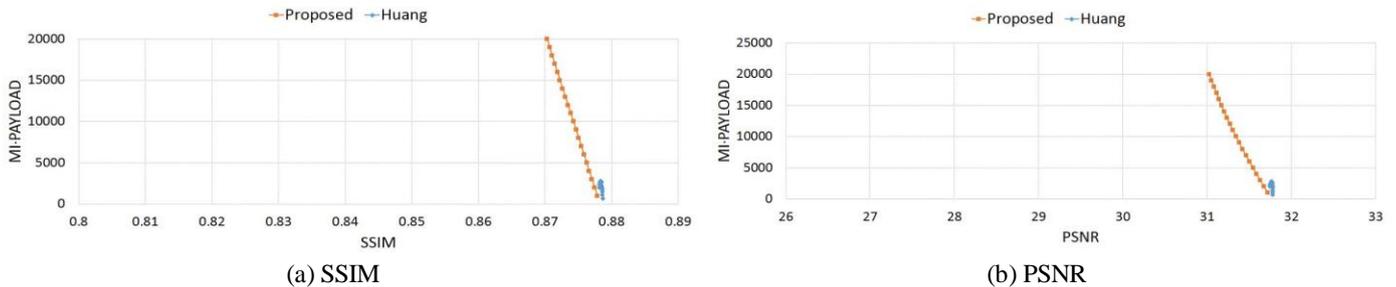

| (a) SSIM | (b) PSNR |

Fig. 6 Visual quality of restored images VS cover images

The results of visual quality described by SSIM and PSNR for five methods are shown in Figs. 4-6. The Y-axis is the multiply of MI and payload: *MI payload*. We use SSIM and PSNR as the X-axis to show for same visual quality, which method can achieve larger useful capacity. This evaluation of combining the influence of visual quality, the accuracy and the capacity. Since only our method and Huang's method are reversible, the other three methods are not shown for restoring images. The MI and useful capacity of our method outperforms Huang's method while the value of visual quality is similar. From the results, one can

see that the proposed method has much better *MI payload* than four competing methods with same visual quality. The visual quality of our proposed method is mainly concentrated in bigger value of SSIM and PSNR, which outperforms four competing methods. With the increment of payload, our proposed method shows a more outstanding and stable performance of the MI and visual quality than these four methods. Our method won't be influenced by the unstable blocks. Thus, MI and visual quality are stable with the payload growing. In conclusion, our method has great and stable performance on visual quality and MI.

5. Conclusions

Many previous literatures noticed that Facebook had huge potential on sharing secret messages. In this work, we performed an in-depth investigation of the various lossy operations on Facebook based on the experimental data. Based upon such prior knowledge, we built a Facebook message sharing system called FMSS that can effectively embed a large amount of data and successfully extract them out from the downloaded images, defeating the uncontrolled lossy operations on Facebook. The experiments revealed that the proposed method offers much higher embedding capacity, and can extract the data successfully with very high probability compared with the state-of-the-art techniques. Furthermore, the restored image upon data extraction is of high quality, and the file size expansion is negligible.

References

[1] A. Castiglione, G. Cattaneo, and A. De Santis, "A forensic analysis of images on online social networks," International Conf. Intelligent Networking and Collaborative Systems (INCoS), IEEE Press, January 2011.

[2] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in jpeg images," IEEE Transactions on Circuits and Systems for Video Technology, vol. 26, no. 9, pp. 1610-1621, September 2016.

[3] J. Hiney, T. Dakve, K. Szczypiorski, and K. Gaj, "Using facebook for image steganography," International Confer. Availability, Reliability and Security (ARES), IEEE Press, August 2015.

[4] A. Castiglione, B. D'Alessio, and A. De Santis, "Steganography and secure communication on online social networks and online photo sharing," International Conf. Broadband and Wireless Computing, Communication and Applications (BWCCA), IEEE Press, October 2011.

[5] S. Nagaraja, A. Houmansadr, P. Piyawongwisal, V. Singh, P. Agarwal, and N. Borisov, "Stegobot: a covert social network botnet," International Workshop on Information Hiding, Springer Berlin Heidelberg, 2011.

[6] K. Solanki, A. Sarkar, and B. Manjunath, "Yass: Yet another steganographic scheme that resists blind steganalysis," International Workshop on Information Hiding. Springer Berlin Heidelberg, pp. 16-31, 2007.

[7] J. Ning, I. Singh, H. V. Madhyastha, S. V. Krishnamurthy, G. Cao, and P. Mohapatra, "Secret message sharing using online social media," IEEE Conf. Communications and Network Security (CNS), IEEE Press, October 2014.

[8] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," Computer, vol. 31, no. 2, pp. 26-34, February 1998.

[9] N. D. Amsden, L. Chen, and X. Yuan, "Transmitting hidden information using steganography via facebook," International Conf. Computing, Communication and Networking Technologies (ICCCNT), IEEE Press, July 2014.

[10] A. Latham, "Steganography," http://linux01.gwdg.de/alatham/stego.html [online]. Accessed 11, October 2016.

[11] G. Schaefer and M. Stich, "Ucid - an uncompressed colour image database," Proceedings of SPIE, vol. 5307, pp. 472-480, 2003.

[12] G. K. Wallace, "The jpeg still picture compression standard," IEEE transactions on consumer electronics, vol. 38, no. 1, pp. xviii-xxxiv, February 1992.

[13] T. Nakahashi and T. Kinoshita, Variable length image coding system, U.S. Patent, 5,319,457, June 07, 1994,