# A Generative Adversarial Network Framework for JPEG Anti-Forensics

Jianyuan Wu*, Li Liu†, Xiangui Kang*, and Wei Sun**
* Guangdong Key Laboratory of Information Security Technology, Sun Yat-sen University, Guangzhou, China
† Kwai Inc., San Jose 95123, USA
** Information Technology Key Laboratory of the Ministry of Education, Sun Yat-sen University, Guangzhou, China

***Abstract*—JPEG anti-forensics aims to remove the artifacts left by JPEG compression and recover JPEG compressed images. However, the existing JPEG anti-forensic methods often introduce new traces and cause the degradation of visual quality of the processed images. In this work, JPEG anti-forensics are modelled as an image-to-image translation problem, where a generative adversarial network framework is used to translate a JPEG compressed image to a reconstructed one. Since JPEG compression causes impairment to high-frequency components, a loss function of high-frequency Discrete Cosine Transform (DCT) coefficients is proposed to recover these components. To prevent forensic detection, a calibration loss function is further introduced to mitigate the variance gap in the high-frequency subbands between generated images and their calibrated versions. Our experimental results demonstrate that the proposed method achieves better image quality than the existing state-of-the-art JPEG anti-forensic methods with comparable anti-forensic performance.**

***Index Terms*—JPEG anti-forensics, GAN, high-frequency loss, calibration loss.**

## I. INTRODUCTION

With recent advances in information technology and the widespread availability of digital equipment, a large number of digital images are recorded and shared every day. To reduce the storage space and transmission time, lossy image compression methods, such as JPEG, are introduced.

It's known that JPEG compression introduces blocking, blurring and ringing artifacts in the spatial domain, and quantization artifacts in the DCT domain. In JPEG [1], quantization is performed according to the standard quantization tables. Since the human visual system is less sensitive to distortion of the high-frequency components smaller quantization steps are adopted for low-frequency term and the high-frequency term is quantized with relatively large steps. As a result, the distortion in high-frequency components is larger than that in low-frequency in JPEG compressed images. Artifacts from JPEG compression in DCT or spatial domain can be used by JPEG forensic algorithms [2, 3] to identify the presence of possible falsifying operations. The goal of JPEG anti-forensics is to conceal or remove the features related to compressed traces extracted by forensic algorithms, thus achieving the purpose of deceiving such algorithms. Stamm et al. [4] proposed a method which can remove quantization artifacts by adding proper noise dither to an image's DCT coefficients. Stamm et al. [5] extended their work by introducing median filtering and white Gaussian noise to disguise blocking artifacts. Fan et al. [6] developed a four-step manipulation process, including total variation (TV)-based deblocking, perceptual DCT histogram smoothing, second-round TV-based deblocking and decalibration, to erase the fingerprints left by JPEG compression in the spatial and DCT domains, achieving a tradeoff between artifact removal and image quality of manipulated images. Fan's method is currently regarded as the benchmark for JPEG anti-forensic performance. However, these anti-forensic methods may introduce new traces, with the noise dithering or filtering operation making the quality of reconstructed anti-forensic images worse than JPEG compressed image. Luo et al. [7] has since pioneered the use of generative adversarial networks (GANs) [8] for JPEG anti-forensics, although this method performs poorly against traditional JPEG forensic methods [9–11].

In this work, we model JPEG anti-forensic efforts as an image-to-image translation problem, and propose a generative adversarial network framework to translate JPEG compressed images into reconstructed ones automatically. Based on the observation that JPEG compression causes serious impairment to high-frequency components, a loss function of high-frequency DCT coefficients is proposed to restore the high-frequency components and improve the visual quality of the reconstructed JPEG images. To prevent forensic detection, a calibration loss is further introduced to mitigate the variance gap in the high-frequency subbands between generated images and their calibrated versions.

## II. NETWORK ARCHITECTURE

Corresponding authors: Xiangui Kang (E-mail: isskxg@mail.sysu.edu.cn), Wei Sun (E-mail: sunwei@mail.sysu.edu.cn). This work was supported by NSFC (Grant nos. 61772571, 62072484) and Chinese national key research and development project 2019QY2203

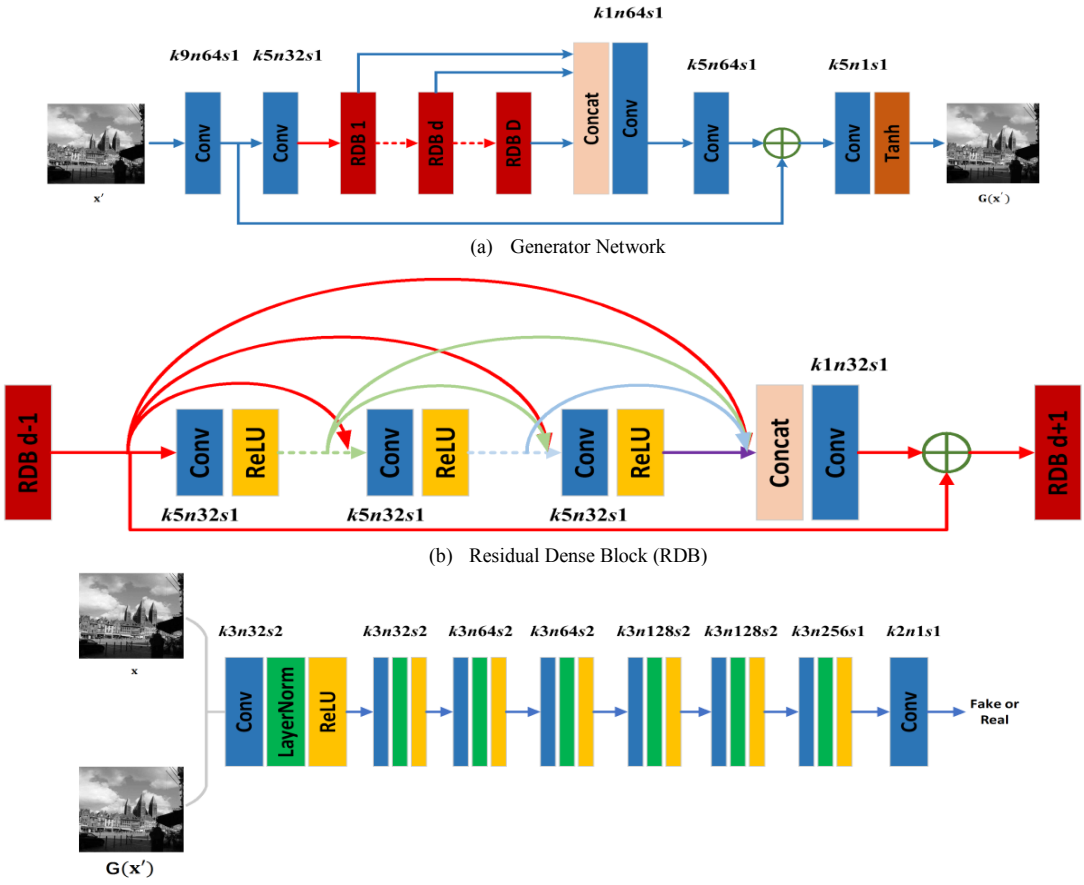



Fig. 1 The architecture of the JPEG anti-forensic framework. The parameters *k*, *n*, and *s* represent the kernel size, the number of kernels and the stride of each convolutional layer, respectively. *G*(**x'**), **x**, and **x'** denote the generated, uncompressed, and JPEG compressed images, respectively.

The architecture of the framework includes a generator network (**G**) and a critic network (**C**), as illustrated in Fig. 1.

### *A. Generator Network*

The architecture of **G** is shown in Fig. 1 - (a). Here we adopt residual dense block (RDB) and residual learning [13]. Firstly, two convolutional layers are used to extract low-level features from JPEG compressed images that are input to the generator. The pixel values of these JPEG image are normalized to [-1, 1]. Eight RDBs are then utilized to learn local dense features, and the output of each RDB is combined directly in a concatenation way. After this concatenation operation, we apply a convolutional layer with 64 1 × 1 kernels to learn new associations across these local feature maps, followed by a convolutional layer with 64 5×5 kernels to extract global features. After adding the preceding feature maps and the output of the first convolutional layer, the resulting feature maps are fed into a convolutional layer with a 5 × 5 kernel. A hyperbolic tangent (TanH) activation function is followed to scale the pixel values of generated image in the range [-1, 1]. It's worth noting that the sizes of the feature maps keep the same in each convolutional layer, because padding mode is set to be the same and the stride equals to 1.

RDB architecture is shown in Fig. 1 - (b). The output of the (*d* - 1)-th RDB is used as the input of the *d*-th RDB. Each RDB consists of six convolutional layers with 32 5 × 5 kernels. Each convolutional layer is followed by rectified linear units (ReLU) [14]. The outputs of the (*d* -1)-th RDB and each of 5 preceding convolutional layers are concatenated to all following layers in the *d*-th RDB. Next, we use a convolutional layer with 32 1 × 1 kernels. The output of this 1 × 1 convolutional layer and the (*d*-1)-th RDB are summed to obtain the output of the *d*-th RDB.

### *B. Critic Network*

**C**, shown in Fig. 1 - (c), is designed to widen the distribution gap of between the generaed images and the original ones. **C** receives either a generated image or an original one. As before, the pixel values of input images are normalized to [–1, 1]. A series of convolutional layers are then employed to learn higher-level typical features. Each convolutional layer with 3 × 3 kernel is followed by a layer normalization [15] and a ReLU activation function. Instead of Batch normalization (BN) [16], layer normalization is used to avoid the interdependence between the input samples of the same batch [17]. The number of filters doubles every two convolutional layers, except for the last layer with a 2 × 2 kernel. In the training process, we used

128 × 128 pixel training images, yielding 2 × 2 feature maps after the six convolutional layers. The critic network ultimately outputs a scalar, representing the score of the input sample. Larger score indicates a greater likelihood that the input image is an original (real) image. Lower score indicates reconstructed or generated (fake) image.

### C. Loss Function of $\boldsymbol{G}$

The loss function in **G** can be represented as:

$$L_{\boldsymbol{G}} = \mathbb{E}_{\mathbf{x}'}\left[\alpha L_G^{pixel} + \beta L_G^{hfc} + \eta L_G^{cal} + \gamma L_G^{adv}\right], \quad (1)$$

where $\mathbf{x}'$ represents the JPEG compressed images. $L_G^{pixel}$, $L_G^{hfc}$, $L_G^{cal}$, and $L_G^{adv}$ represent the pixel-wise loss, high-frequency loss, calibration loss, and adversarial loss, respectively. $\alpha$, $\beta$, $\eta$, and $\gamma$ refer to the pre-defined weights for each loss term. The training goal for **G** is to obtain optimal model parameters and minimize $L_G$, *i.e.*, solving the optimization problem of $\min_{\boldsymbol{\theta_G}} L_G$. $\boldsymbol{\theta_G}$ denotes the parameters for **G**.

1) *Pixel-Wise Loss*

Given an uncompressed image $\mathbf{x}$ and its corresponding JPEG compressed image $\mathbf{x}'$, both with the size of $W \times H$, the pixel-wise mean squared error loss $L_G^{pixel}$ is defined as:

$$L_G^{pixel} = \frac{1}{S}\sum_{i=1}^{S} ||\mathbf{x} - G(\mathbf{x}')||_2^2, \quad (2)$$

where $\|\cdot\|_2^2$ means the square of $l_2$ norm; $S = W \times H$; and $G(\mathbf{x}')$ represents a generated image. The pixel-wise loss reflects the differences between the pixels of an original image and the output image of **G**. The optimization of $L_{\boldsymbol{G}}^{pixel}$ leads to a high PSNR value of the output image from **G** [18].

2) *High-frequency Loss*

A high-frequency loss $L_G^{hfc}$ is proposed to restore the high-frequency components of JPEG images. It is defined as an $\ell_2$ loss (i.e., a Euclidean distance) with respect to the differences in the high-frequency DCT coefficients of the generated image and its original image. It can be represented as:

$$L_G^{hfc} = \frac{1}{N}\sum_{k=1}^{N} ||DCT(\mathbf{x}^k)\odot\mathbf{M} - DCT(G(\mathbf{x}')^k)\odot\mathbf{M}||_2^2, \quad (3)$$

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ \vdots & \iddots & \vdots & \iddots & 1 \\ 0 & 0 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 & 1 \end{bmatrix}, \quad (4)$$

where $N = S/64$ represents the number of blocks; the superscript $k$ refers to the block index of an image; $\odot$ means dot product of matrices; and $DCT(\cdot)$ denotes the DCT transformation of 8 × 8 blocks which do not overlap in the image. As shown in Eq. (4), **M** is a mask matrix where each element in the upper-left corner along the counter-diagonal of **M** equals to zero and all other elements are 1. **M** can be regarded as an ideal high-pass filter in the DCT domain. $L_G^{hfc}$ provides an effective representation of the loss of high-frequency details in the DCT domain, and leads to the generation of visually realistic images.

3) *Calibration Loss*

For an original uncompressed image, the variance of the DCT coefficients is very similar to that of its calibrated version in the high-frequency subbands [19], which can be obtained by cropping four pixels at the top, bottom, left and right edges of the original uncompressed image. However, for a JPEG compressed image, the variance of DCT coefficients is quite different from that of its calibrated version in the 28 high-frequency subbands. Furthermore, even after anti-forensic attack [4, 5], the gap of the variance of DCT coefficients between the anti-forensically modified image and the original one is still large.

To achieve anti-forensic performance, the proposed calibration loss function is introduced to minimize the variance gap in the high-frequency subbands between the generated image and its calibrated version, which can be defined as:

$$L_{\boldsymbol{G}}^{\text{cal}} = \frac{1}{28}\sum_{j=1}^{28}\Big|\text{var}\left(block\_DCT_j\left(G(\mathbf{x}')\right)\right) - \text{var}\left(block\_DCT_j(G(\mathbf{x}')_{cal})\right)\Big|, \quad (5)$$

where $block\_DCT_j(\cdot)$ represents all DCT coefficients of the $j$-th high-frequency subband extracted after 8 × 8 DCT transformation of the image, with the locations of the values of 1 in matrix **M** (see Eq. (4)) corresponding to 28 high-frequency subbands; $\text{var}(\cdot)$ refers to the variance of the input vector; $G(\mathbf{x}')_{cal}$ indicates the calibrated version of a generated image; and $|\cdot|$ means the absolute value.

4) *Adversarial Loss*

To generate images with statistical characteristics similar to the original images, we also need to employ the adversarial loss. From the perspective of **G**, we expect images generated by **G** to deceive **C** as far as possible. Therefore, we define the adversarial loss $L_G^{adv}$ as:

$$L_G^{adv} = -C\left(G(\mathbf{x}')\right), \quad (6)$$

where $C(\cdot)$ refers to the output of the critic network.

### D. Loss Function of $\boldsymbol{C}$

**G** and **C** are trained iteratively. **G** is fixed when **C** is trained, and vice versa. **C** is trained to maximize the gap between the distributions of the original and generated images, i.e., the loss function of **C** is defined as [17]:

$$L_C = \mathbb{E}_{\mathbf{x}}[C(\mathbf{x})] - \mathbb{E}_{\mathbf{x}'}\left[C\left(G(\mathbf{x}')\right)\right] + \lambda\mathbb{E}_{\hat{\mathbf{x}}}[(\|\nabla_{\hat{\mathbf{x}}}C(\hat{\mathbf{x}})\|_2 - 1)^2] \quad (7)$$

where $\hat{\mathbf{x}} = \epsilon\mathbf{x} + (1-\epsilon)G(\mathbf{x}')$ denotes random samples sampled uniformly from $\mathbf{x}$ and $G(\mathbf{x}')$; $\epsilon \sim U[0, 1]$ is a uniform distributed random number; $\|\cdot\|_2$ means the $\ell_2$ norm; and $\lambda$ is the gradient penalty coefficient constant set to 10 [17].

## III. Experimental Results

The proposed framework was implemented in TensorFlow [20] and trained on a workstation equipped with an Nvidia GTX TITAN XP GPU. In all experiments, we used three public image datasets: BossBase V1.01 (BossBase), [21] BOWS2-Original (BOWS) [22], and UCID-V2 (UCID) [23]. We randomly extracted 8,000 uncompressed images from

Table I. The average PSNR and SSIM values of JPEG compressed images $\mathcal{J}$ and the anti-forensically modified images with $\mathcal{STL}$, $\mathcal{SL}$ $\mathcal{FX}$, $\mathcal{LK}$, $\mathcal{A}$. The anti-forensic performance (AUC values) against the detectors in [9-12] is also evaluated.

| QF | Methods | Lai [9] | Luo [12] | Fan [10] | Val [11] | PSNR (dB) | SSIM |
|---|---|---|---|---|---|---|---|
| 25 | $\mathcal{J}$ | 1.00 | 1.00 | 0.98 | 0.97 | 32.59 | 0.8950 |
| | $\mathcal{STL}$ [4] | 1.00 | 0.14 | 1.00 | 0.87 | 29.00 | 0.6911 |
| | $\mathcal{SL}$ [5] | 0.92 | 0.10 | 0.81 | 0.88 | 30.00 | 0.7995 |
| | $\mathcal{FX}$ [6] | 0.34 | 0.25 | **0.50** | **0.50** | 32.32 | 0.8876 |
| | $\mathcal{LK}$ [7] | 0.99 | 0.66 | 0.91 | 0.93 | 32.89 | 0.8973 |
| | $\mathcal{A}$ | **0.54** | **0.57** | 0.72 | 0.22 | **33.36** | **0.9046** |
| 50 | $\mathcal{J}$ | 1.00 | 1.00 | 0.97 | 0.98 | 35.07 | 0.9346 |
| | $\mathcal{STL}$ [4] | 1.00 | 0.15 | 0.99 | 1.00 | 31.18 | 0.7741 |
| | $\mathcal{SL}$ [5] | 0.72 | 0.12 | 0.65 | 0.91 | 31.15 | 0.8848 |
| | $\mathcal{FX}$ [6] | 0.32 | 0.28 | **0.50** | **0.52** | 34.52 | 0.9233 |
| | $\mathcal{LK}$ [7] | 0.99 | 0.69 | 0.83 | 0.87 | 34.97 | 0.9315 |
| | $\mathcal{A}$ | **0.42** | **0.57** | 0.67 | 0.09 | **35.21** | **0.9350** |

BossBase and BOWS for training. Each image was randomly cropped into 9 non-overlapped 128 × 128 images, each three of which were compressed using the MATLAB JPEG compressor with a quality factor (QF) of 25, 50, and 75, respectively. The training dataset therefore had 8,000 × 2 × 9 = 144,000 compressed images and corresponding uncompressed images. Only images with the size of 128 × 128 were used for training. The remaining 4,000 images from BossBase and BOWS with the size of 512 × 512, together with 1,338 uncompressed gray images of size 384 × 512 converted from UCID, were used as the testing images.

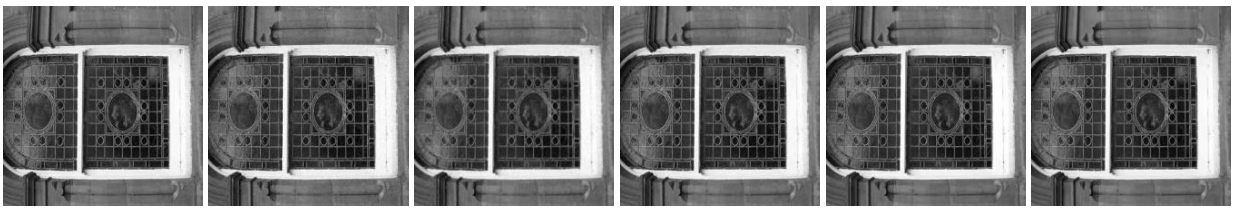
The proposed GAN networks are trained to their convergence after 85,000 iterations. In the first 10,000 iterations, only **G** was trained and $\gamma$ was set to zero to avoid the critic network focusing on the image content rather than on whether or not the image had been compressed. In the latter 75,000 iterations, **C** and **G** were trained once alternately. The weights of loss functions in Eq. (1) are set as: $\alpha = 1.0$, $\beta = 0.1$, $\eta = 50.0$, and $\gamma = 1.0 \times 10^{-5}$. We set a fixed learning rate of $1.0 \times 10^{-4}$ for both the generator and the critic network. The batch size is set to 16 and the optimizer Adam [24] is adopted with the default setting.

### *A. Anti-Forensics of JPEG Compression*

In order to conceal the quantization artifacts of the DCT coefficients [11], the JPEG compressed images are pre-processed by adding noise dither to each DCT coefficient using the $\mathcal{STL}$ method in [4]. Then the networks are trained using the pre-processed images as **G**'s input. Finally, the trained **G** is used to generate the modified images with anti-forensic enhancements. This scheme is denoted as $\mathcal{A}$ in this work.

1) *Countering the Conventional Forensic Detectors*

We compared the anti-forensic performance of the proposed anti-forensic method $\mathcal{A}$ with four existing JPEG anti-forensic methods $\mathcal{STL}$ [4], $\mathcal{SL}$ [5], $\mathcal{FX}$ [6], and $\mathcal{LK}$ [7]. As mentioned in the end of Para. 1 of Subsection *A*, the remaining 4,000 images from BossBase and BOWS with the size of 512 × 512, together with 1,338 uncompressed gray images of size 384 × 512 converted from UCID, were used as the testing images. For each anti-forensic method and each JPEG QF in {25, 50}, each test image was JPEG compressed and anti-forensically modified. Then we used 5,338 anti-forensically modified images as positive samples and 5,338 uncompressed test images as negative samples to obtain the area under the receiver operating characteristic curve (AUC) in the detection with the detectors in Lai [9], Fan [10], Valenzise [11], and Luo [12] respectively. If the AUC value is close to or less than 0.5, the anti-forensics method is effective in deceiving the forensic



(a) $\mathcal{J}$ (b) $\mathcal{STL}$ [4] (c) $\mathcal{SL}$ [5] (d) $\mathcal{FX}$ [6] (e) $\mathcal{LK}$ [7] (f) $\mathcal{A}$

Fig. 2 Visual examples of various JPEG anti-forensic methods. (a) JPEG compressed image $\mathcal{J}$; (b) – (f) $\mathcal{J}$ anti-forensically modified with $\mathcal{STL}$, $\mathcal{SL}$, $\mathcal{FX}$, $\mathcal{LK}$ and the proposed method $\mathcal{A}$, respectively.

Table II. Detection error rates (%) of the CNN-based forensic detector BS [26] in detecting whether a compressed image/an anti-forensically modified image has been compressed.

| | QF=25 | QF=50 | QF=75 |
|---|---|---|---|
| $\mathcal{J}$ | 0.02 | 0.10 | 2.67 |
| $\mathcal{STL}$[4] | 22.92 | 80.20 | 95.35 |
| $\mathcal{SL}$[5] | **100.00** | **100.00** | **100.00** |
| $\mathcal{FX}$[6] | 98.65 | 97.20 | 97.00 |
| $\mathcal{LK}$[7] | 95.38 | 98.23 | 99.95 |
| $\mathcal{A}$ | 94.55 | 90.23 | 95.88 |

detector. As shown in Table I, the proposed method $\mathcal{A}$ is very close to or less than 0.5, and has comparable performance with state-of-the-art anti-forensic method $\mathcal{FX}$, while achieving better image visual quality in terms of the peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [25] than all existing anti-forensic methods.

Fig. 2 shows the visual examples of images modified with the various anti-forensic methods. It can be seen that our method $\mathcal{A}$ improved the image quality. Compared with the other JPEG anti-forensic methods, the reconstructed image obtained with $\mathcal{A}$ had better clarity and more complete details. Our method appeared more natural to the human eye. $\mathcal{STL}$, $\mathcal{SL}$, and $\mathcal{FX}$ sacrificed the image quality of the JPEG image, and introduced a certain degree of noise to the image, making the processed image more blurred and reducing its visual appeal.

2) *Countering CNN-Based Forensic Detector*

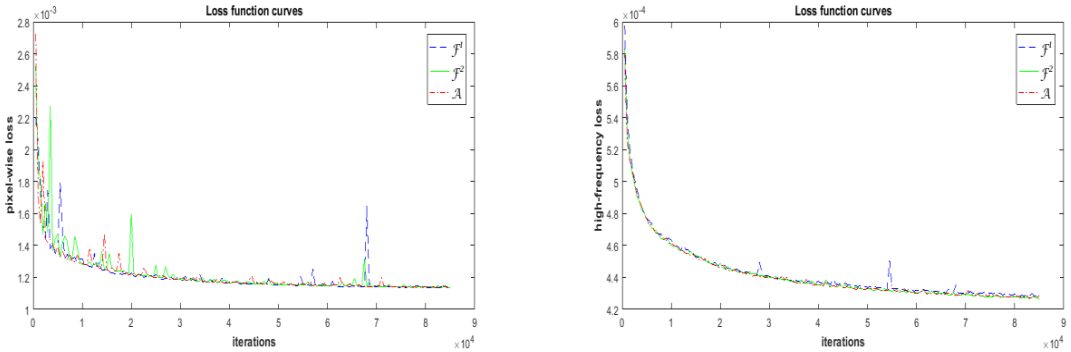
Bayar et al. [26] proposed a forensic detector (BS) based on convolution neural network (CNN) and it is trained using pairs of original uncompressed images and their JPEG compressed counterparts. More specifically, the training dataset included 16,000 512 × 512 images from BossBase and BOWS. Each original uncompressed image was JPEG compressed with a QF randomly chosen from {25, 50, 75}. The trained detector is used to detect whether the compressed and anti-forensically modified images had been compressed. The test dataset included the remaining 4,000 original images from the BossBase and BOWS datasets with size of 512 × 512. For each JPEG QF and each anti-forensic method, each image in the test dataset was JPEG compressed and anti-forensically modified, then it is detected whether it is uncompressed (original).

As shown in Table II, $\mathcal{SL}$, $\mathcal{FX}$, $\mathcal{LK}$ and the proposed method anti-forensic method $\mathcal{A}$ have the excellent anti-forensic performance for all three JPEG QFs, with a detection error rate closing to 1.0, meaning that almost all anti-forensically modified image was detected as being uncompressed (original) images. The proposed $\mathcal{A}$ also showed comparable performance in deceiving BS with $\mathcal{SL}$, $\mathcal{FX}$, $\mathcal{LK}$.

### *B. Ablation Study*

To investigate the interactions among the parts of the loss function of **G** in the anti-forensics of JPEG images, two additional models were trained, one containing only $L_G^{pixel}$, represented here by $\mathcal{F}^1$; and the other containing $L_G^{pixel}$ and $L_G^{hfc}$, denoted here as $\mathcal{F}^2$. $\mathcal{F}^1$ and $\mathcal{F}^2$ had the same training parameters as $\mathcal{A}$. The variation in $L_G^{pixel}$ and $L_G^{hfc}$ was then observed per 500 iterations during the training of $\mathcal{A}$, $\mathcal{F}^1$, and $\mathcal{F}^2$ on the validation set, which is composed of a total of 16,014 JPEG compressed images with size 128×128 and the corresponding uncompressed original images. Each three of JPEG images were obtained by randomly cropping the original testing image into three non-overlapped images and then carrying out JPEG compression with QF 25, 50, and 75, respectively. Fig. 3 shows the convergence of the models and the varying curves of $L_G^{pixel}$ and $L_G^{hfc}$ for the validation set.

From Fig. 3 - (a), it can be seen that the values of $L_G^{pixel}$ for $\mathcal{A}$, $\mathcal{F}^1$, and $\mathcal{F}^2$ decreased and converged during training. The locations of the convergence of $L_G^{pixel}$ are basically the same in the later stages of the training process, and there is no obvious difference among them. It can therefore be considered that adding $L_G^{hfc}$ or $L_G^{adv}$ to the total loss function of **G** has little influence on the final convergence position of $L_G^{pixel}$. On the other hand, it can be seen from Fig. 3 - (b) that the values of $L_G^{hfc}$ corresponding to $\mathcal{A}$, $\mathcal{F}^1$, and $\mathcal{F}^2$ also decreased and converged during training. The $L_G^{hfc}$ curve of $\mathcal{F}^1$ shows that if only $L_G^{pixel}$ is optimized, $L_G^{hfc}$ will be reduced. Since the




(a) $L_G^{pixel}$ (b) $L_G^{hfc}$

Figure 3. Loss function curves of $L_G^{pixel}$ and $L_G^{hfc}$ during the training of $\mathcal{A}$, $\mathcal{F}^1$, and $\mathcal{F}^2$ on the validation set.

purpose of optimizing $L_G^{pixel}$ is to bring the pixel values of the reconstructed image closer to those of reference image, the low- and high-frequency components of the JPEG image will be restored simultaneously in the early stages of network training. However, by comparing the $L_G^{hfc}$ curve of $\mathcal{F}^1$ with that of $\mathcal{F}^2$ and $\mathcal{A}$ in the later stages of training, we can see that the $L_G^{hfc}$ curves of $\mathcal{F}^2$ and $\mathcal{A}$ lie primarily below that of $\mathcal{F}^1$, indicating that if the $L_G^{hfc}$ part is added to the total loss function of **G**, the convergence position of $L_G^{hfc}$ in the later stages of training with $\mathcal{F}^2$ or $\mathcal{A}$ will be smaller than that of $L_G^{hfc}$ where **G** optimizes only $L_G^{pixel}$. This demonstrates that in the late training stage, loss of the high-frequency components of reconstructed images accounts for a larger proportion of the total loss than in the early training period. If $L_G^{hfc}$ is added to the total loss function of **G** as a supplement to $L_G^{pixel}$, **G** will be guided to focus more on restoring the high-frequency components of a JPEG image in the later training period, thus the reconstruction of an over-smoothed image can be avoided.

## IV. CONCLUSIONS

This work explores the removal of artifacts left by JPEG compression and proposes a GANs framework for reconstructing JPEG compressed images. We propose a loss function of high-frequency DCT coefficients to recover high-frequency components impaired by JPEG compression and to reconstruct JPEG compressed images with statistical characteristics and visual quality similar to the original uncompressed images. In order to prevent forensic detection, we propose a calibration loss to mitigate the variance gap in the high-frequency subbands between generated images and their calibrated versions. Our theoretical analysis and experiments show that the proposed loss functions accurately restore JPEG compressed images and enhance the JPEG anti-forensic performance. The experimental results demonstrate that the proposed JPEG anti-forensic method achieves a better tradeoff between avoiding forensic detection and preserving image quality than state-of-the-art JPEG anti-forensic methods.

## REFERENCES


[1] B. Pennebaker and L. Mitchell, JPEG still image data compression standard. New York, NY, USA: Van Nostrand Reinhold, 1993.

[2] Z. Fan and R. de Queiroz. Identification of bitmap compression history: JPEG detection and quantizer estimation. IEEE Transactions on Image Processing, 12(2): 230–235, 2003.

[3] T. Bianchi and A. Piva. Detection of nonaligned double JPEG compression based on integer periodicity maps. IEEE Transactions on Information Forensics and Security, 7(2): 842–848, 2012.

[4] M. C. Stamm, S. K. Tjoa, W. S. Lin, and K. J. R. Liu. Anti-forensics of JPEG compression. In Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 1694–1697, 2010.

[5] M. C. Stamm and K. J. R. Liu. Anti-forensics of digital image compression. IEEE Transactions on Information Forensics and Security, 6(3):1050–1065, 2011.

[6] W. Fan, K. Wang, F. Cayre, and Z. Xiong. JPEG anti-forensics with improved tradeoff between forensic undetectability and image quality. IEEE Transactions on Information Forensics and Security, 9(8):1211–1226, 2014.

[7] Y. Luo, H. Zi, Q. Zhang, and X. Kang. Anti-forensics of JPEG compression using Generative Adversarial Networks. In Proceeding of European Signal Processing Conference, pp. 957–961, 2018.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems (NIPS), pp. 2672–2680, 2014.

[9] S. Lai and R. Böhme. Countering counter-forensics: The case of JPEG compression. In Proc. of 5th ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec), pp. 285–298, 2011.

[10] W. Fan, K. Wang, F. Cayre, and Z. Xiong. A variational approach to JPEG anti-forensics. In Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 3058–3062, 2013.

[11] G. Valenzise, M. Tagliasacchi, and S. Tubaro. Revealing the traces of JPEG compression anti-forensics. IEEE Transactions on Information Forensics and Security, 8(2):335–349, 2013.

[12] W. Luo, J. Huang, and G. Qiu. JPEG error analysis and its applications to digital image forensics. IEEE Trans. Inf. Forensics Security, vol. 5, no. 3, 2010, 480–491.

[13] Y. Zhang, Y. Tian, Y Kong, B. Zhong, and Y. Fu. Residual Dense Network for Image Super-Resolution. In Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2472–2481, 2018.

[14] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In Proc. of 27th International Conference on Machine Learning (ICML), pp. 807–814, 2010.

[15] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

[16] S, Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.0316, 2015.

[17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein GAN. In advances in Neural Information Processing Systems (NIPS), 2017, pp. 5767-5777.

[18] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. IEEE Transactions on Computational Imaging, 3(1):47–57, 2017.

[19] E.Y. Lam and J. Goodman. A mathematical analysis of the DCT coefficient distributions for images. IEEE Transactions on Image Processing, 9(10):1661–1666, 2000.

[20] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al. TensorFlow: A system for large-scale machine learning. In Proc. of 12th USENIX Conference on Operating Systems Design and Implementation (OSDI), pp. 265–283, 2016.

[21] P. Bas, T. Filler, and T. Pevny. Break our steganographic system—The ins and outs of organizing BOSS. In Proc. of the 13th International Conference on Information Hiding (IH), pp. 59–70, 2011.

[22] P. Bas and T. Furon. Break Our Watermarking System. [Online]. Available: http://bows2.ec-lille.fr, 2007.

[23] G. Schaefer and M. Stich. UCID—An uncompressed colour image database, In Proceedings of SPIE, 2004, 472–480.

[24] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In Proc. of International Conference on Learning Representations (ICLR), 2015.

[25] A. Hore and D. Ziou. Image quality metrics: PSNR Vs. SSIM. In Proc. of International Conference on Pattern Recognition (ICPR), pp. 2366–2369, 2010.

[26] B. Bayar and M. C. Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, pp.. 5–10, 2016.


.