A robust document image watermarking scheme using deep neural network

Sulong Ge\textsuperscript{1,2} · Zhihua Xia\textsuperscript{1,2,3} · Jianwei Fei\textsuperscript{1,2} · Yao Tong\textsuperscript{4} · Jian Weng\textsuperscript{3} · Ming Li\textsuperscript{3}

Received: 22 April 2022 / Revised: 2 September 2022 / Accepted: 27 February 2023 /
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
Watermarking is an important copyright protection technology which generally embeds the identity information into the carrier imperceptibly. Then the identity can be extracted to prove the copyright from the watermarked carrier even after suffering various attacks. Most of the existing watermarking technologies take the nature images as carriers. Different from the natural images, document images are not so rich in color and texture, and thus have less redundant information to carry watermarks. This paper proposes an end-to-end document image watermarking scheme using the deep neural network. Specifically, an encoder and a decoder are designed to embed and extract the watermark. A noise layer is added to simulate the various attacks that could be encountered in reality, such as the Cropout, Dropout, Gaussian blur, Gaussian noise, Resize, and JPEG Compression. A text-sensitive loss function is designed to limit the embedding modification on characters. An embedding strength adjustment strategy is proposed to improve the quality of watermarked image with little loss of extraction accuracy. Experimental results show that the proposed document image watermarking technology outperforms three state-of-the-art methods in terms of the robustness and image quality.

Keywords Watermark · Document image · Noise layer · Deep neural network

1 Introduction

The development of computer multimedia technology provides great conveniences for e-commerce and e-government, in which many valuable documents, such as administrative documents, certificates, medical cases and transaction certificates, are scanned to be the document images in digital format. The document images can be stored and exchanged efficiently but can also be illegally copied and stolen by unauthorized persons. In addition to the encryption technologies for the confidentiality, watermarking techniques are generally designed to protect the copyright of image [9]. As shown in Fig. 1, the identity information of data owner can be embedded as the watermark $w$ into a cover document image

Extended author information available on the last page of the article.
In this paper, we propose a robust document image watermarking scheme by using DNN. The main contributions can be concluded as follows:

- To the best of our knowledge, the proposed watermarking method is the first DNN-based one for document images. A text-sensitive loss function is designed to decrease the modification on text characters. The noise layer is constructed to simulate various attacks to improve the robustness. The watermark expansion strategy also helps to increase the robustness.

- It is found that the visual effect of the watermarked image is unsatisfying although the Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM) seem good enough, which can be attributed to the clear background of the document image. Accordingly, we proposed an embedding strength adjustment strategy to increase the image quality with little loss of extraction accuracy.

- Due to the lack of ready-made document image dataset, we construct two large-scale document image datasets for the DNN training. One is named DocImgEN which includes 230,000 training, 10,000 validation and 10,000 testing document images with
English sentences. The other is named DocImgCN including 230,000 training, 10,000 validation and 10,000 testing document images with Chinese sentences.

The rest of the paper is organized as follows. The related works are presented in Section 2. The proposed scheme is described in Section 3. The experiment results and analysis are presented in Section 4. Finally, Section 5 draws the conclusion of our work.

2 Related works

In this section, we firstly introduce the watermarking technologies for document images. Then several DNN-based watermarking technologies are discussed as we will use DNN to construct our method.

2.1 Watermarking methods for document images

The existing document image watermarking methods can be divided into two categories: structure-based and image-based ones.

The structure-based methods This kind of methods embed watermarks into document images by exploiting the specific structure in text documents. Brassil et al. [5, 6] are the first to study document image watermarking. Three technologies, i.e., line-shift coding, word-shift coding, and character coding, are proposed and discussed. The line-shift coding and word-shift coding in [5, 6] require the original unmarked document image for watermark extraction and get the low embedding capacity. The character feature encoding also requires the original document for extraction and the watermark can be easily affected by local noise since it marks on local features [13]. Accordingly, Huang and Yan [13] proposed a document image watermarking method to achieve a blind extraction. The watermark is associated to a sine wave with the specific phase and frequency. The average inter-word spaces of lines in document are adjusted according to the sine wave. This method supports blind watermark extraction but also have low embedding capacity. Kim et al. [18] also proposed a document image watermarking method by adjusting the inter-word spaces. Firstly, the words in the document are categorized into different classes according to width of its adjacent words. Then, the segment, consisting of several adjacent words, can be also classified according to the class of the words in it. Finally, the watermark bits are embedded by modifying statistics of inter-word spaces in segment classes. This method embeds the same watermark bits in each class of segments and is robust even though some words or segments are missed. Amano and Misaki [3] proposed a document image watermarking method by changing the width of strokes. Specifically, a text area, such as a line of words, is divided into two separate parts. Then, according to the watermark bit, the character strokes in one part are changed to be fatter and that in the other part are changed to be thinner. During the watermark extraction, the average width of the strokes in two parts are compared to figure out the bit. Tan et al. [29] also proposed a document image watermarking method based on strokes of Chinese characters. The watermark bits are embedded by modulating the direction of strokes, and the shuffling is used to balance embedding payload.

The image-based approaches This kind of methods process the document image as a whole during the watermarking. Kim et al. [17] applied the Sobel edge operator to generate the edge direction histogram. Authors revealed that the normalized edge direction
histograms generated from the document image blocks in the same language are quite constant. Thus, the document image can be divided into blocks. Some of blocks are chosen as the reference blocks while the others can be slightly modified to adjust its edge direction histogram according to the watermark bits. Loc et al. [22] stated that the layout of a document could be quite complex and proposed to divide the document image using full convolution networks. Then, watermark bits are embedded into the appropriate segments. Specifically, each appropriate image segment is divided into blocks. If all the pixel values in a block are larger than a threshold, the block will be further divided into two parts. The watermark bits are embedded by adjusting the average of the pixel values in two parts. Lu et al. [23] proposed a watermarking technology for the binary image in discrete cosine transform (DCT) domain. For the satisfying imperceptibility and robustness to binarization, the binary images are firstly blurred to be the gray ones by Gaussian filter. Then, the gray image is divided into $8 \times 8$ blocks and the non-uniform blocks are transformed into DCT domain. The DC component is modified to carry the watermark bits. Finally, the whole gray image is binarized back. This method is proved to be robust to print and scan operations but the watermark extraction needs the participation of the original image. Horng et al. [27] also proposed a document image watermarking method in DCT domain. Firstly, the image is divided into $8 \times 8$ blocks and then transformed by DCT. Next, the $8 \times 8$ blocks are decomposed by singular value decomposition (SVD). Then the singular values are adjusted to embed the watermark bits. Chetan and Nirmala [8] proposed to watermark document images in Discrete Wavelet Transform (DWT) domain. The document image is divided into segments and the non-empty segments are transformed by Level-2 DWT. Then the watermark bits are embedded in the LL2 sub-bands. Al-Haj and Barouqa [2] further proposed a watermarking method in DWT domain by using SVD. Similar to [27], the selected sub-band is decomposed by SVD and the singular values are adjusted to embed the watermark bits. Dang et al. [10] proposed a watermarking scheme to embed a Quick Response (QR) Code into the HH2 sub-band of the DWT domain. The coding strategy of QR Code can provide extra robustness to the noise. Barouqa and Al-Haj [4] proposed a document image watermarking method by using DWT and the Schur decomposition (SD). The authors transform the cover into transform domain by Level-2 DWT, then divide the HH2 sub-band into the same infixed-size blocks, and finally apply the SD to embed watermark bits on each block. This method has good robustness to Gaussian noise attacks.

Generally, the structure-based document image watermarking technologies have good robustness but low capacity. On the contrary, the image-based ones hold higher capacity but are not so robust to the noises.

### 2.2 DNN-based watermarking methods for natural images

The Deep Neural Network (DNN) designed for watermarking generally includes an encoder and a decoder for watermark embedding and extraction, respectively. Besides, the loss functions related to the image quality and watermark error are designed to train the encoder and decoder.

Mun et al. [26] proposed a robust watermarking method using DNN. The cover images are divided into non-overlapping blocks, and the encoder and decoder are designed to embed and extract the watermark in each block. To enhance the robustness, noises are added into the watermarked images which are then inputted into the decoder for reinforcement training. Considering that the watermarking in transform domain is more secure and robust against attacks, Ahmadi et al. [1] proposed an embedding network structure including two transform layers. One layer is employed to transform the cover image into transform domain before
the encoder while the other is used to transform the concatenation of cover and watermark back. Zhong et al. [34] proposed to add an Invariance Layer (IL) between the encoder and decoder. A regularization term is employed in IL to preserve useful information related to the watermark, while removing all other noise and irrelevant information. Zhu et al. [35] proposed a noise layer to simulate various kinds of attacks in reality. During the training stage, the watermarked images from the encoder are input into the noise layer with random parameters. Then, the noised images are fed into the decoder for watermark extraction. Liu et al. [21] stated that the one-stage end-to-end training can converge slowly and receive low-quality watermarked images due to the noise attack, and thus proposed a two-stage deep learning watermarking scheme. The first stage conducts a noise-free end-to-end training and the second refines the decoder with noise attacks. Luo et al. [25] stated that noise layer in [21, 35] were the differentiable models and could generalize poorly to unknown distortions. The authors added a generative adversarial network (GAN) between the encoder and decoder to generate distortion. In addition, channel coding is designed to add redundancy to the watermark, which also increases the robustness.

In addition to above methods, invertible neural network (INN) as a novel DNN has been used to design the watermarking method, in which the encoding and decoding process share the same network framework. In the forward hiding process, the cover image and watermark are encoded to generate a watermarked image, and in the backward decoding process, the watermarked image is decoded to recover the watermark. In these methods, to keep the same size of input and output dimension, auxiliary information needs to be introduced. Jing et al. [15] proposed an image watermarking method by using INN. To make better hiding performance, they decomposed the cover and watermark image into frequency domain by DWT before feeding into the forward hiding process. In the backward concealing stage, inverse wavelet transform (IWT) is used to output the watermarked image. At this base, Guan et al. [12] proposed a multiple images watermarking method by using INN. To guide the multiple images hiding, the authors design an important map (IM) module in the forward hiding process, and importance loss is also designed to cooperate the module use. In addition, multi-stage training strategy is developed to guide the multiple images hiding, which increases the visual quality of watermarked image.

3 The proposed document image watermarking scheme

In this section, we firstly give an overview of our scheme. Then, the encoder, noise layer, decoder, and loss function are specified. Besides, we also present the acronyms used in the paper in Table 1.

| Acronym | Description |
|---------|-------------|
| I_c     | cover image |
| w       | watermark   |
| I_w     | watermarked image |
| I_w'    | noised and watermarked image |
| w'      | recovered watermark |
| w_e     | expanded watermark |
| m_w     | watermark mask |
3.1 The overview of the proposed scheme

Inspired by the DNN-based watermarking methods for natural images [1, 21, 25, 26, 34, 35], we propose an end-to-end watermarking scheme for document images. As illustrated in Fig. 2, the Encoder \( E \) and Decoder \( D \) are constructed to embed and extract watermarks, a Noise Layer (NL) is designed to simulate the possible distortion in reality, and the loss functions are calculated to optimize the encoder and decoder by considering the particularity of document images.

3.2 Encoder

The Encoder \( E \) is a network trained to embed the watermark into cover document image. Firstly, the watermark \( w \) is expanded to bring in redundancy. Then the expanded watermark and cover are encoded together, generating a watermark mask \( m_w \). Finally, the watermark mask is added to the cover to generate the watermarked image \( I_w \).

Watermark expansion In our scheme, the watermark \( w \) can be an arbitrary string of binary bits. Before encoded with the cover, the watermark \( w \) is expanded to bring in redundancy through a fully connected layer, which helps to enhance the robustness to noise attacks. Then the extended watermark is reshaped and upsampled to be a three-dimensional tensor \( w_e \) with the same size as \( I_c \).

Encoding The expanded watermark \( w_e \) and the cover document image \( I_c \) are concatenated and encoded through convolutional layer Conv1-7 as illustrated in Fig. 3, generating a watermark mask \( m_w \). The DNN-based watermarking method is essentially using the convolutional maps to co-encode with the watermarks [21]. Thus, it is better to learn the watermarking mode by using convolutional maps of different levels. Accordingly, as illustrated in Fig. 3, the expanded watermark \( w_e \) is concatenated to the output of Conv1-5, and the cover document image \( I_c \) is concatenated to the output of Conv2 and Conv5.

Addition The convolutional layers output a three-channel watermark mask \( m_w \) which is finally added to the cover as follow,

\[
I_w = I_c + \alpha \cdot m_w, \tag{1}
\]

where \( \alpha \) is an embedding strength factor.

![Fig. 2](image-url) The overview of the proposed scheme. In the training stage, the watermark \( w \) is expanded, reshaped, and concatenated with the cover image \( I_c \). Then, combination of \( I_c \) and \( w \) are encoded to generate the watermarked \( I_w \). Next, \( I_w \) is input into the noise layer which simulates the attacks on input, generating the noised and watermarked image \( I'_w \). Finally, \( I'_w \) is input into the Decoder \( D \) to extract the watermark \( w' \) which could be similar or the same to \( w \). After training, the Encoder \( E \) is used to embed the watermark and the Decoder \( D \) is used for watermark extraction.
3.3 Noise layer

The watermarked image can suffer various distortions during the storage and transmission. The watermarking scheme cannot be robust to these attacks without appropriate training. Accordingly, a Noise Layer (NL) is added between the encoder and decoder in the training stage to bring in distortions, such as Dropout, Cropout, Gaussian Blur, Gaussian Noise, Resize, and JPEG Compression, as illustrated in Fig. 4. The attacks in reality are expected to be the individual or combination of these distortions.

- **Dropout.** A part of pixels in $I_w$ is randomly selected and replaced by the pixels in $I_c$ at the corresponding positions. The ratio of selected pixels is denoted as $r_d$ and we set $r_d \leq 10\%$.
- **Cropout.** The pixels in a randomly chosen region of $I_w$ keep unchanged, and the rest is replaced by the corresponding part of $I_c$. The ratio of the unchosen region to the whole $I_w$ is denoted as $r_c$ and we set $r_c \leq 10\%$.
- **Gaussian Blur:** A Gaussian blur is conducted on $I_w$ with the window size of $\delta_b \times \delta_b$ and a random standard deviation from $[1.0, 3.0]$.
- **Gaussian noise.** Gaussian noise with a random standard deviation $\delta_n \in [0, 0.02)$ is generated and added to $I_w$.
- **Resize:*** The size of watermarked image $I_w$ is reduced and then amplified to the original size. The reduction ratio $r_r$ is randomly chosen from $[0, 50\%]$.
- **JPEG Compression:** JPEG Compression includes several steps such as color space conversion, block splitting, discrete cosine transformation, quantization, and so on. Among these steps, the quantization cannot be directly incorporated into the training network as it is not differentiable. Thus, we use the method in $[24, 28]$ to approximate the effect of quantization step as,

\[
d_i \leftarrow \left[ \left( \frac{d_i}{M_i^f \times q_f} \right) + \left( \left( \frac{d_i}{M_i^f \times q_f} - \frac{d_i}{M_i^f \times q_f} \right) \right)^3 \cdot (M_i^f \times q_f) \right],
\]

(2)

![Fig. 3](image.png) The structure of the encoder $E$

![Fig. 4](image.png) The structure of Noise Layer $NL$
The distortions above are illustrated in Fig. 5.

### 3.4 Decoder

The Decoder $D$ is trained to extract the watermark ($w'$) from the watermarked image. In the training stage, the input of $D$ is the noisy watermarked image $I_w'$. Inspired by AlexNet [20], the decoder consists of seven convolutional layers, a flatten layer, and a dense layer, as illustrated in Fig. 6. The stride size of the convolutional layer Conv1 and Conv3 is set to be the default value 1, while the others are set to 2 to speed up the training. After Conv7, a flatten operation is performed on the remaining neurons, and then a dense layer is used to output an 1-dimensional tensor which has the same size of $w$. Sigmoid function is used to produce a output in the last layer.
3.5 Loss function

The loss function in our scheme consists of three parts: image loss, text-sensitive loss, and watermark loss.

**Image loss** Image loss is designed to keep the watermarked image $I_w$ similar to the cover $I_c$. We consider the image in YUV color space and try to make little change on the Y component as the human eyes are more sensitive to it. In addition, the large modification on a single pixel can be more destructive to visual effect. Accordingly, we restrain the big change on pixels. The image loss is designed as

$$L_I = \text{MSE}(I_w^Y, I_c^Y) \times s_Y + \text{MSE}(I_w^U, I_c^U) \times s_U + \text{MSE}(I_w^V, I_c^V) \times s_V,$$

where $\text{MSE}$ denotes the mean squared error, and $s_Y, s_U$ and $s_V$ are the weights for YUV channels.

**Text-sensitive loss** Readers will pay more attention to the characters during reading. The modification on characters can be more conspicuous [11, 33]. Thus, the text-sensitive loss is designed to restrain the modification on characters as follows,

$$L_T = |I_w^R - I_c^R| \cdot \tilde{I}_c^R \times s_R + |I_w^G - I_c^G| \cdot \tilde{I}_c^G \times s_G + |I_w^B - I_c^B| \cdot \tilde{I}_c^B \times s_B,$$

where $\tilde{I}_c^* = \frac{255 - I_c^*}{255}$ assigns larger punishment to the dark points, and $s_*$ denotes the weights for different color components, $* \in \{R, G, B\}$.

**Watermark loss** Watermark loss is designed to keep the extracted watermark as similar as the original. Binary cross entropy function is used for it as

$$L_W = - \sum_{i=1}^{N} (w_i \cdot \log(w_i^f) + (1 - w_i) \cdot \log(1 - w_i^f)),$$

where $w_i$ refers to the original watermark, $w_i^f$ denotes the extracted watermark and $N$ is the bit number of the watermark.

Finally, the total training loss is calculated as

$$L_{total} = \lambda_I L_I + \lambda_T L_T + \lambda_W L_W,$$

where $\lambda_I, \lambda_T,$ and $\lambda_W$ are weight factors.

4 Experimental results and analysis

This section presents our self-made datasets, implementation details, and experimental results. Besides, we discuss an embedding strength adjustment strategy which increases the image quality without much loss of extraction accuracy.
4.1 The construction of document image datasets

Unlike the traditional document watermarking technologies, the DNN-based method needs to be trained with the large-scale training dataset. However, we regrettably find that there are currently no such datasets. Accordingly, we construct two large-scale document image datasets: DocImgEN and DocImgCN, and the examples are shown in Fig. 7. Please note that, the datasets can be downloaded from https://github.com/gslxr/Document-image-watermarking for research.

**DocImgEN**

We download a batch of PDFs from IEEE Xplore database [14] and convert the PDF pages to JPEG images. Then, the image blocks with the size $400 \times 400$ are cropped out as the document images. DocImgEN includes 230,000 training, 10,000 validation and 10,000 testing document images with English words.

**DocImgCN**

We download PDFs from China National Knowledge Infrastructure (CNKI) [7] to prepare document images with Chinese characters. Similarly, DocImgCN also includes 230,000 training, 10,000 validation and 10,000 testing document images.

4.2 Implementation details

We implement and test our scheme on TensorFlow with a GPU: NV ADIA GTX 1080Ti. The parameters in our scheme are summarized in Table 2. The first 100,000 training images in DocImgEN and DocImgCN are utilized to train the corresponding models. Four images and four random bit strings assemble a training batch. The learning rate is set to be 0.0001 and Adam optimizer [19] is employed to optimize the models. All the testing images in DocImgEN and DocImgCN are used to verify the performance.

During the training stage, it is found that sometimes the decoder failed to achieve a satisfying extraction accuracy even with a large number of training iterations. Thus, we froze the encoder at the first 3000 iterations, just training the decoder and noisy layer. In this setting, the encoder and decoder are fine-tuned by gradient descent. In this way, the encoder can be pre-trained and the decoder can be fine-tuned.

**DocImgEN**

| Training image | Validation image | Testing image |
|----------------|------------------|--------------|
| ![Image](image1.png) | ![Image](image2.png) | ![Image](image3.png) |

**DocImgCN**

| Training image | Validation image | Testing image |
|----------------|------------------|--------------|
| ![Image](image4.png) | ![Image](image5.png) | ![Image](image6.png) |

Fig. 7 The examples of the document images in DocImgEN and DocImgCN datasets.
Table 2  The setting of parameters in the training stage

| Parameter | $r_d$ | $r_c$ | $r_r$ |
|-----------|-------|-------|-------|
| Range     | (0, 10%) | (0, 10%) | [0, 50%] |
| Parameter | $\delta_b$ | $\delta_n$ | $q$ |
| Range or value | 3 | (0, 0.02] | [50, 100) |
| Parameter | $s_Y$ | $s_U$ | $s_V$ |
| Value     | 100 | 1 | 1 |
| Parameter | $s_R$ | $s_G$ | $s_B$ |
| Value     | 3 | 6 | 1 |
| Parameter | $\lambda_I$ | $\lambda_T$ | $\lambda_W$ |
| Value     | 1.5 | 1.5 | 2.0 |

addition, to make the decoder gradually adapt to the noise distortion, the parameters $\lambda_I$, $\lambda_T$, and $\lambda_W$ are set to be 0 at the beginning, and increase linearly to 1.5, 1.5, and 2.0 at the first 15,000 iterations.

4.3 The robustness of the proposed scheme

In our scheme, the noise layer is an important strategy to improve the robustness of the watermark. The tests are conducted on DocImgEN and DocImgCN separately. The watermark length here is 100 bits. Please note that, as listed in Table 2, we perform the distortions with a relatively low intensity in noise layer during the training stage, so as to guarantee the quality of the watermarked image $I_w$. After training, the trained models are tested with higher intensity distortions to verify the robustness.

Here we test our scheme in three different cases. At first, the encoder and decoder are trained without the noise layer. The generated model in this case is named as basic model. Next, the encoder and decoder are trained with the noise layer that just considers a single distortion. Accordingly, six models are generated for six types of distortions per each dataset, respectively, and named as specified models. Finally, the encoder and decoder are trained with the noise layer that considers all six distortions together. The resulting model is named as combined model. The training iterations for the basic, specified, and combined models are set to be 50,000, 60,000, and 80,000, respectively, which are large enough for adequate training. The results for DocImgEN and DocImgCN are shown in Figs. 8 and 9.

As shown in Figs. 8 and 9, although noise layer is not incorporated in training, the basic model is still robust to the distortions to some extent, especially to Gaussian blur and resize. This robustness derives from the DNN structure in encoder and decoder. Next, the specified model that considers a single distortion is tested by the identical distortion with various intensity. Substantial improvements are achieved. Most importantly, the combined model gets better robustness than the specified models for all six distortions, especially with the higher distortion intensity. It indicates that the encoder and decoder in our scheme are powerful enough to learn appropriate embedding strategies for resisting various distortions. Besides, the comprehensive consideration of various distortions can improve robustness to each single distortion. The results in following are generated by the combined model.

4.4 The quality of the watermarked document image

The visual quality of the watermarked image is guaranteed by the image loss and text-sensitive loss. Two common measurements, i.e., Peak Signal to Noise Ratio (PSNR) and
Structural Similarity Metric (SSIM) [31], are utilized to evaluate the similarity between the cover and watermarked images. Specifically, PSNR measures the similarity in pixel level while SSIM calculates similarity from the brightness, contrast, and structure. Considering that readers may be more sensitive to the characters while reading the documents, we designed the text-sensitive loss to ensure less modification on text pixels. Here we define
the Change Intensity Per Text-Pixel (CPP) to evaluate the modification on text pixels as follows,

\[
CPP = \frac{\sum_{i=1}^{n_t} |I_{#c}^R(i) - I_{#w}^R(i)|}{n_t} + \frac{\sum_{i=1}^{n_t} |I_{#c}^G(i) - I_{#w}^G(i)|}{n_t} + \frac{\sum_{i=1}^{n_t} |I_{#c}^B(i) - I_{#w}^B(i)|}{n_t},
\]

where \(I_{#c}^*\), \(I_{#w}^*\), \# \(\in\{c, w\}\), \(*\) \(\in\{R, G, B\}\) refers to the set of text pixels in the cover and watermarked images, and \(n_t\) denotes the total number of the text pixels.
During the testing of image quality, the watermark length is also set to be 100 bits. The PSNR, SSIM, and CPP values are listed in Table 3, and some example pairs of cover and watermarked images are shown in Fig. 10. The results are averaged from all testing images in DocImgEN and DocImgCN testsets. As listed in Table 3, the incorporation of text-sensitive loss $L_T$ make little influence on PSNR and SSIM but decrease CPP by 57.52% averagely. It indicates that $L_T$ decreases the modification on characters without much influence on image quality.

4.5 Comparison with the state-of-the-art methods

We compared our scheme with three DNN-based watermarking models, i.e., HiDDeN [35], Liu et al. [21] and Stegasstamp [30], in terms of robustness and image quality. As the three models are trained for nature images, we retrained these end-to-end models on DocImgEN and DocImgCN. The length of the watermark is 100 bits and the rest training parameters.
Table 4  The PSNR, SSIM, and CPP of schemes on the DocImgEN and DocImgCN

| Schemes          | PSNR (dB) | SSIM | CPP  |
|------------------|-----------|------|------|
| HiDDeN [35] (On DocImgEN) | 33.80     | 0.937 | 14.75 |
| Liu et al. [21] (On DocImgEN) | 27.20     | 0.891 | 39.89 |
| Stegastamp [30] (On DocImgEN) | 34.79     | 0.977 | 7.43  |
| Ours (On DocImgEN) | 40.10     | 0.962 | 3.52  |
| HiDDeN [35] (On DocImgCN) | 33.70     | 0.939 | 12.47 |
| Liu et al. [21] (On DocImgCN) | 28.20     | 0.901 | 32.15 |
| Stegastamp [30] (On DocImgCN) | 35.70     | 0.984 | 8.35  |
| Ours (On DocImgCN) | 41.07     | 0.965 | 3.77  |

are set as that in original paper [21, 30, 35]. Please note that, the noise distortions added in the noise layer are the same as ours.

The bit accuracy is calculated under different distortions and listed in Tables 5 and 6. It shows that our scheme achieves the best robustness. The schemes in [21, 30, 35] are designed for nature images while our scheme is specially designed for the document image. In addition, some effective operations in [21, 30, 35] are kept in our model such as the noise layer, watermark expansion, and concatenation of watermark during the convolution process. As listed in Table 4, our scheme also achieves the best PSNR and CPP while Stegastamp [30] holds better SSIM than ours. The perceptual loss [32] could be helpful for high SSIM. Finally, the watermarked images by [21, 30, 35] are illustrated in Fig. 11.

Fig. 11  The examples of watermarked images from our scheme and the state-of-the-art methods
### Table 5  Bit accuracy of schemes on the DocImgEN.

| Attack type       | HiDDeN [35] | Liu et al. [21] | Stegstamp [30] | Ours  |
|-------------------|-------------|-----------------|----------------|-------|
| Dropout ($r_d=10\%$) | 93.33       | 96.34           | 99.96          | 100   |
| Dropout ($r_d=30\%$) | 84.76       | 86.25           | 99.92          | 100   |
| Dropout ($r_d=50\%$) | 75.85       | 74.73           | 99.73          | 99.99 |
| Cropout ($r_c=10\%$) | 93.06       | 82.26           | 99.93          | 100   |
| Cropout ($r_c=30\%$) | 83.86       | 76.21           | 99.83          | 99.95 |
| Cropout ($r_c=50\%$) | 82.16       | 75.47           | 98.57          | 98.48 |
| Gaussian blur ($\delta_b=3$) | 95.13       | 98.88           | 99.99          | 100   |
| Gaussian blur ($\delta_b=5$) | 94.72       | 98.75           | 99.99          | 100   |
| Gaussian blur ($\delta_b=7$) | 94.23       | 98.57           | 99.97          | 100   |
| Gaussian noise ($\delta_n=0.02$) | 94.92       | 98.27           | 99.84          | 100   |
| Gaussian noise ($\delta_n=0.03$) | 93.31       | 96.88           | 99.68          | 100   |
| Gaussian noise ($\delta_n=0.05$) | 89.93       | 92.01           | 98.11          | 99.83 |
| Resize ($r_r=50\%$) | 94.40       | 98.70           | 99.99          | 100   |
| Resize ($r_r=30\%$) | 91.61       | 97.85           | 99.91          | 99.99 |
| Resize ($r_r=10\%$) | 79.20       | 78.32           | 78.96          | 97.53 |
| JPEG compression ($q=50$) | 90.17       | 93.76           | 99.68          | 99.81 |
| JPEG compression ($q=30$) | 85.14       | 91.85           | 92.91          | 96.62 |
| JPEG compression ($q=20$) | 80.73       | 87.40           | 86.29          | 88.68 |

### Table 6  Bit accuracy of schemes on the DocImgCN

| Attack type       | HiDDeN [35] | Liu et al. [21] | Stegstamp [30] | Ours  |
|-------------------|-------------|-----------------|----------------|-------|
| Dropout ($r_d=10\%$) | 98.20       | 100             | 99.67          | 100   |
| Dropout ($r_d=30\%$) | 90.68       | 99.92           | 99.44          | 99.96 |
| Dropout ($r_d=50\%$) | 77.40       | 94.18           | 99.22          | 99.90 |
| Cropout ($r_c=10\%$) | 97.49       | 99.55           | 99.74          | 99.99 |
| Cropout ($r_c=30\%$) | 84.67       | 82.89           | 99.15          | 99.68 |
| Cropout ($r_c=50\%$) | 79.79       | 77.01           | 98.31          | 97.99 |
| Gaussian blur ($\delta_b=3$) | 99.13       | 100             | 99.68          | 100   |
| Gaussian blur ($\delta_b=5$) | 98.90       | 100             | 99.66          | 100   |
| Gaussian blur ($\delta_b=7$) | 98.69       | 100             | 99.58          | 100   |
| Gaussian noise ($\delta_n=0.02$) | 97.10       | 99.99           | 99.45          | 99.95 |
| Gaussian noise ($\delta_n=0.03$) | 94.10       | 99.85           | 99.35          | 99.91 |
| Gaussian noise ($\delta_n=0.05$) | 89.84       | 97.06           | 98.45          | 99.64 |
| Resize ($r_r=50\%$) | 98.70       | 100             | 99.70          | 100   |
| Resize ($r_r=30\%$) | 97.86       | 100             | 99.68          | 99.99 |
| Resize ($r_r=10\%$) | 88.70       | 83.85           | 97.58          | 97.73 |
| JPEG compression ($q=50$) | 97.87       | 98.97           | 97.52          | 99.72 |
| JPEG compression ($q=30$) | 91.91       | 94.59           | 92.46          | 95.24 |
| JPEG compression ($q=20$) | 80.19       | 81.23           | 81.40          | 81.32 |
Table 7 The PSNR, SSIM and CPP of our scheme with different embedding strength during the testing stage

| Schemes                  | PSNR (dB) | SSIM  | CPP   |
|--------------------------|-----------|-------|-------|
| On DocImgEN with $\alpha=1.0$ | 40.10     | 0.962 | 3.52  |
| On DocImgEN with $\alpha=2.0$ | 44.80     | 0.990 | 2.07  |
| On DocImgEN with $\alpha=3.0$ | 46.45     | 0.993 | 2.05  |
| On DocImgEN with $\alpha=4.0$ | 46.80     | 0.995 | 2.58  |
| On DocImgEN with $\alpha=5.0$ | 48.30     | 0.996 | 2.46  |
| On DocImgCN with $\alpha=1.0$ | 41.07     | 0.965 | 3.77  |
| On DocImgCN with $\alpha=2.0$ | 44.97     | 0.986 | 2.95  |
| On DocImgCN with $\alpha=3.0$ | 45.23     | 0.992 | 2.91  |
| On DocImgCN with $\alpha=4.0$ | 46.90     | 0.995 | 2.01  |
| On DocImgCN with $\alpha=5.0$ | 47.30     | 0.996 | 2.60  |

4.6 Further improvements by adjusting embedding strength

Although the PSNR and SSIM values in Table 4 indicate a satisfying quality with the 100 bits of watermark, the examples in Fig. 11 show clear embedding traces at the background of document images. It can be attributed to the clean background of document image. We tried to alleviate this phenomenon by adjusting the embedding strength in training and testing stages.

Fig. 12 The visual examples of our scheme with different embedding strength during the testing stage
The adjustment of the embedding strength $\alpha$ will influence the quality of watermarked image and the bit accuracy according to the (1). As shown in Tables 5 and 6, our scheme achieves good bit accuracy even after the attacks with high strength. It indicates a space to decrease the embedding strength $\alpha$, so as to improve the quality of the watermarked image without much loss in bit accuracy. That is to say, we can set a relatively high embedding strength during the training process. After the model is fully trained, we can decrease the embedding strength in real application.

In this subsection, we try to set the embedding strength $\alpha = 1.0, 2.0, 3.0, 4.0, 5.0$ during the training process and decrease $\alpha$ to be 1.0 in the testing stage. As shown in Table 7 and Fig. 12, with larger initial $\alpha$, better image quality is achieved, while as shown in Tables 8 and 9, the bit accuracy is not decreased a lot except for the JPEG-compressed images.

## 5 Conclusions

This paper proposed an end-to-end document image watermarking scheme using deep neural network. An encoder and a decoder are designed to embed and extract the watermark. A noise layer is incorporated to simulate the various attacks such as the Cropout, Dropout, Gaussian Blur, Gaussian Noise, Resize, and JPEG Compression. Watermark is expanded to increase the robustness. The cover image and expanded watermark are repeatedly concatenated with the tensor during the training process to improve the performance. A text-sensitive loss function is designed to decrease the embedding modification on characters. An embedding strength adjustment strategy is further proposed to decrease the embedding trace with little loss of robustness. The extensive experiments demonstrated

| Attack type          | $\alpha=1.0$ | $\alpha=2.0$ | $\alpha=3.0$ | $\alpha=4.0$ | $\alpha=5.0$ |
|----------------------|--------------|--------------|--------------|--------------|--------------|
| Dropout ($r_d=10\%$) | 100          | 99.92        | 99.93        | 99.69        | 99.86        |
| Dropout ($r_d=30\%$) | 100          | 99.74        | 99.68        | 99.10        | 99.28        |
| Dropout ($r_d=50\%$) | 99.99        | 98.42        | 98.37        | 97.03        | 96.79        |
| Cropout ($r_c=10\%$) | 100          | 99.98        | 99.94        | 99.91        | 99.91        |
| Cropout ($r_c=30\%$) | 99.95        | 99.30        | 99.15        | 98.89        | 98.80        |
| Cropout ($r_c=50\%$) | 98.48        | 96.03        | 95.53        | 94.50        | 92.86        |
| Gaussian blur ($\delta_b=3$) | 100         | 100          | 100          | 99.96        | 99.96        |
| Gaussian blur ($\delta_b=5$) | 100         | 100          | 100          | 99.96        | 99.96        |
| Gaussian blur ($\delta_b=7$) | 100         | 100          | 100          | 99.95        | 99.94        |
| Gaussian noise ($\delta_n=0.02$) | 100        | 99.36        | 99.21        | 98.29        | 97.57        |
| Gaussian noise ($\delta_n=0.03$) | 100         | 97.76        | 97.28        | 94.51        | 92.81        |
| Gaussian noise ($\delta_n=0.05$) | 99.83     | 90.92        | 90.44        | 83.99        | 78.80        |
| Resize ($r_r=50\%$) | 100          | 100          | 100          | 99.99        | 99.99        |
| Resize ($r_r=30\%$) | 99.99        | 100          | 99.99        | 99.97        | 99.92        |
| Resize ($r_r=10\%$) | 97.53        | 97.34        | 97.10        | 89.63        | 91.95        |
| JPEG compression ($q=50$) | 99.81    | 89.17        | 88.28        | 76.99        | 73.94        |
| JPEG compression ($q=30$) | 96.62   | 69.77        | 67.86        | 63.66        | 42.06        |
| JPEG compression ($q=20$) | 88.68  | 42.43        | 38.89        | 38.02        | 30.44        |
Table 9  Bit accuracy of our scheme with the different embedding strength during the testing stage on DocImgCN

| Attack type           | $\alpha$=1.0 | $\alpha$=2.0 | $\alpha$=3.0 | $\alpha$=4.0 | $\alpha$=5.0 |
|----------------------|--------------|--------------|--------------|--------------|--------------|
| Dropout ($r_d=10\%$) | 100          | 100          | 99.61        | 99.31        | 99.24        |
| Dropout ($r_d=30\%$) | 99.96        | 99.94        | 99.21        | 98.52        | 98.09        |
| Dropout ($r_d=50\%$) | 99.90        | 99.53        | 97.33        | 95.41        | 94.04        |
| Cropout ($r_c=10\%$) | 99.99        | 100          | 99.68        | 99.58        | 99.58        |
| Cropout ($r_c=30\%$) | 99.68        | 99.63        | 98.45        | 98.52        | 97.97        |
| Cropout ($r_c=50\%$) | 97.99        | 96.28        | 92.44        | 93.07        | 92.78        |
| Gaussian blur ($\delta_b=3$) | 100          | 100          | 100          | 100          | 99.97        |
| Gaussian blur ($\delta_b=5$) | 100          | 100          | 99.99        | 100          | 99.97        |
| Gaussian blur ($\delta_b=7$) | 100          | 100          | 99.99        | 100          | 99.97        |
| Gaussian noise ($\delta_n=0.02$) | 99.95        | 99.89        | 98.98        | 95.69        | 95.13        |
| Gaussian noise ($\delta_n=0.03$) | 99.91        | 99.38        | 97.08        | 88.35        | 89.47        |
| Gaussian noise ($\delta_n=0.05$) | 99.64        | 94.98        | 87.83        | 71.16        | 76.56        |
| Resize ($r=50\%$) | 100          | 100          | 99.98        | 99.97        | 99.97        |
| Resize ($r=30\%$) | 99.99        | 99.98        | 99.97        | 99.91        | 99.95        |
| Resize ($r=10\%$) | 97.73        | 87.35        | 90.32        | 86.60        | 91.41        |
| JPEG compression ($q=50$) | 99.72        | 85.23        | 79.79        | 69.46        | 64.84        |
| JPEG compression ($q=30$) | 95.24        | 68.09        | 61.62        | 57.83        | 46.16        |
| JPEG compression ($q=20$) | 81.32        | 45.74        | 41.13        | 41.09        | 40.42        |

the superiority of our scheme. In future work, it could be promising to incorporate the embedding strength adjustment strategy in watermarking scheme for nature images.

Acknowledgements  This work is supported in part by the National Key Research and Development Plan of China under Grant number 2022YFB3103100, 2020YFB1005600, in part by Guangdong Basic and Applied Basic Research Foundation under Grant number 2019B1515120010, in part by the National Natural Science Foundation of China under grant numbers 62122032, 62172233, 62102189, U19361118, 61932011, 61931004, 61825203, U1736203, and 61732021, in part by the Major Program of Guangdong Basic and Applied Research Project under Grant 2019B030302008, in part by Six Peak Talent project of Jiangsu Province (R2016L13), Qinglan Project of Jiangsu Province, and “333” project of Jiangsu Province, in part by the National Joint Engineering Research Center for Network Security Detection and Protection Technology, in part by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund, in part by the Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAAET) fund, China. Zhihua Xia is supported by BK21+ program from the Ministry of Education of Korea.

Data Availability  Some or all data, models, or code generated or used during the study are available in the submitted article.

Declarations

Conflict of Interests  There are no potential conflicts of interest in this work, and no human participants or animals are involved.
References

1. Ahmadi M, Norouzi A, Karimi N, Samavi S, Emami A (2020) Redmark: framework for residual diffusion watermarking based on deep networks. Expert Syst Appl 146:113157

2. Al-Haj A, Barouqa H (2017) Copyright protection of e-government document images using digital watermarking. In: 2017 3rd international conference on information management (ICIM). IEEE, pp 441–446

3. Amano T, Misaki D (1999) A feature calibration method for watermarking of document images. In: Proceedings of the fifth international conference on document analysis and recognition. ICDAR’99 (Cat. No. PR00318). IEEE, pp 91–94

4. Barouqa H, Al-Haj A (2021) Watermarking e-government document images using the discrete wavelets transform and schur decomposition. In: 2021 7th international conference on information management (ICIM). IEEE, pp 102–106

5. Brassil J, Low S, Maxemchuk N, O’Gorman L (1994) Electronic marking and identification techniques to discourage document copying. In: Proceedings of INFOCOM’94 conference on computer communications. IEEE, pp 1278–1287

6. Brassil J, Low S, Maxemchuk N (1999) Copyright protection for the electronic distribution of text documents. Proc IEEE 87(7):1181–1196

7. China national knowledge infrastructure (2021) [Online]. Accessed: April 4. Available: https://www.cnki.net/

8. Chetan KR, Nirmala S (2015) An efficient and secure robust watermarking scheme for document images using integer wavelets and block coding of binary watermarks. J Inf Secur Appl 24:13–24

9. Cox I, Miller M, Bloom J, Fridrich J, Kalker T (2007) Digital watermarking and steganography. Morgan kaufmann, San Mateo

10. Dang QB, Louisa K, Coustaty M, Luqman MM, Ogier J-M (2019) A blind document image watermarking approach based on discrete wavelet transform and qr code transformation. Morgan kaufmann, San Mateo

11. Fang H, Zhang W, Ma Z, Zhou H, Sun S, Cui H, Nenghai Y (2019) A camera shooting resilient watermarking scheme for underpainting documents. IEEE Trans Circuits Syst Video Technol 30(11):4075–4089

12. Guan Z, Jing J, Deng X, Xu M, Jiang L, Zhang Z, Li Y (2022) Deepminh: deep invertible network for multiple image hiding. IEEE Trans Pattern Anal Mach Intell

13. Huang D, Yan H (2001) Interword distance changes represented by sine waves for watermarking text images. IEEE Trans Circuits Syst Video Technol 11(12):1237–1245

14. Ieee/iet electronic library (2021) [Online]. Accessed: April 4. Available: https://ieeexplore.ieee.org/Xplore/home.jsp

15. Jing J, Deng X, Xu M, Wang J, Guan Z (2021) Hinet: deep image hiding by invertible network. In: Proceedings of the IEEE/cvf international conference on computer vision, pp 4733–4742

16. Kamaruddin NS, Kamsin A, Por LY, Rahman H (2018) A review of text watermarking: theory, methods, and applications. Pattern Recogn Lett 25(11):1243–1251

17. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv: 1412.6980

18. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25:1097–1105

19. Liu Y, Guo M, Zhang J, Zhu Y, Xie (2019) A novel two-stage separable deep learning framework for practical blind watermarking. In: Proceedings of the 27th ACM international conference on multimedia, pp 1509–1517

20. Loc CV, Burie J-C, Ogier J-M (2018) Document images watermarking for security issue using fully convolutional networks. In: 2018 24th International conference on pattern recognition (ICPR). IEEE, pp 1091–1096

21. Lu H, Shi X, Shi YQ, Kot AC, Chen L (2002) Watermark embedding in dc components of dct for binary images. In: 2002 IEEE workshop on multimedia signal processing. IEEE, pp 300–303

22. Luo X, Talebi H, Yang F, Elad M, Milanfar P (2021) The rate-distortion-accuracy tradeoff: jpeg case study. In: 2021 Data compression conference (DCC), pp 354–354

23. Luo X, Zhan R, Chang H, Yang F, Milanfar P (2020) Distortion agnostic deep watermarking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 13548–13557
26. Mun Seung-Min, Nam Seung-Hun, Jang H, Kim D, Lee Heung-Kyu (2019) Finding robust domain from attacks: a learning framework for blind watermarking. Neurocomputing 337:191–202
27. Rosiyadi D, Horng S-J, Fan P, Wang X, Khan MK, Pan Y (2011) Copyright protection for e-government document images. IEEE Multimed 19(3):62–73
28. Shin R, Song D (2017) Jpeg-resistant adversarial images. In: NIPS 2017 workshop on machine learning and computer security, vol 1
29. Tan L, Sun X, Sun G (2012) Print-scan resilient text image watermarking based on stroke direction modulation for Chinese document authentication. Radioengineering 21(1)
30. Tancik M, Mildenhall B, Ng R (2020) Stegastamp: invisible hyperlinks in physical photographs. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2117–2126
31. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
32. Zhang R, Isola P, A Efros A, Shechtman E, Wang O (2018) The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 586–595
33. Zhao H, Gallo O, Frosio I, Kautz J (2016) Loss functions for image restoration with neural networks. IEEE Transactions on computational imaging 3(1):47–57
34. Zhong X, Huang Pei-Chi, Mastorakis S, Shih F (2021) An automated and robust image watermarking scheme based on deep neural networks. IEEE Trans Multimed 23:1951–1961
35. Zhu J, Kaplan R, Johnson J, Fei-Fei L (2018) Hidden: hiding data with deep networks. In: Proceedings of the European conference on computer vision (ECCV), pp 657–672

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Sulong Ge
Affiliations

Sulong Ge\(^{1,2}\) · Zhihua Xia\(^{1,2,3}\) · Jianwei Fei\(^{1,2}\) · Yao Tong\(^{4}\) · Jian Weng\(^{3}\) · Ming Li\(^{3}\)

Sulong Ge
13512983683@163.com

Jianwei Fei
fjw826244895@163.com

Yao Tong
melody@fongwell.com

Jian Weng
cryptjweng@gmail.com

Ming Li
limjnu@gmail.com

1 School of Computer, Nanjing University of Information Science & Technology, 219 Ningliu Road, Nanjing, 210044, Jiangsu, China
2 Engineering Research Center of Digital Forensics, Ministry of Education, 219 Ningliu Road, Nanjing, 210044, Jiangsu, China
3 College of Cyber Security, Jinan University, 601 Huangpu Road West, Guangzhou, 510632, Guangdong, China
4 Guangzhou Fongwell Data Limited Company, 371-1 Wushan Road, Guangzhou, 510510, Guangdong, China