High Capacity Reversible Watermarking for Images Based on Classified Neural Network

Rongrong Ni[1,⋆], H.D. Cheng[2], Yao Zhao[1], and Yu Hou[1]

[1] Institute of Information Science,
Beijing Jiaotong University Beijing 100044, China
rrni@bjtu.edu.cn
[2] Department of Computer Science,
Utah State University Logan, Utah, USA

Abstract. Reversible watermarking is a useful technique for some applications requiring high image quality because it can restore what the original images are as well as protect them. In this paper, a high capacity image reversible watermarking is proposed based on classified neural network. According to the variance of surrounding pixel values, all pixel cells are classified as smooth part or rough part. Correspondingly, two neural networks are designed for smooth pixel prediction and rough pixel prediction, respectively. The watermark is embedded in the prediction errors. In addition, a retesting strategy utilizing the parity detection is presented to increase the capacity of the algorithm. Experimental results show that this algorithm can get smaller prediction error and obtain both higher capacity and good visual quality.

Keywords: Reversible watermarking, classified neural network, retesting strategy.

1 Introduction

Reversible watermarking can recover the original digital contents without any distortion after data have been extracted. It has been an active research topic for the applications where the availability of the original content is essential. So far, many schemes have been proposed.

Early reversible watermarking algorithms mainly utilize lossless compression to provide space for data embedding. A more effective algorithm is histogram shifting, introduced by Ni et al.[1], which moves the histogram bars to achieve low distortion. Another productive approach is difference expansion algorithm, which was proposed by Tian [2]. The method divides the image into pairs of pixels and uses each legitimate pair for hiding one bit of information. It has high

⋆ This work was supported in part by 973 Program (2011CB302204), National Natural Science Funds for Distinguished Young Scholar (61025013), National NSF of China (61073159, 61272355), Sino-Singapore JRP (2010DFA11010), Fundamental Research Funds for the Central Universities (2012JBM042).

J.-K. Kämäräinen and M. Koskela (Eds.): SCIA 2013, LNCS 7944, pp. 697–706, 2013.

embedding capacity and good quality, which becomes the basic idea of some reversible watermarking methods. Higher capacity and better visual quality is the main purpose that reversible watermarking methods pursue. Recently, prediction error expansion (PEE) method has been proposed by Thodi et al.[3]. The method uses PEE to embed data, and suggests incorporating expansion embedding with histogram shifting to reduce the location map. Since then, several PEE-based methods have been proposed [4–8]. In [7], Sachnev et al. proposed a method which combined sorting and two-pass-testing with prediction error expansion method. The algorithm obtains higher capacity and lower distortion than most of existing reversible watermarking methods.

In this paper, a high capacity image reversible watermarking is proposed based on classified neural network. Considering the global feature, the neural network is used to predict the prediction error. According to the variance of surrounding pixel values, the pixel cells are classified as smooth part or rough part. Correspondingly, two neural networks are designed for smooth pixel prediction and rough pixel prediction, respectively. The watermark is embedded in the prediction errors. Because the actually embedded data is not always identical with the testing bit, some ambiguous pixel cells appear. A retesting strategy utilizing the parity detection activates the capacity of these ambiguous pixel cells. As a result, the capacity is increased. The experimental results show that this algorithm can obtain higher capacity and preserve good visual quality.

2 Proposed Algorithm Based on Classified Neural Network and Retesting Strategy

In the proposed algorithm, all pixels of the image are divided into two sets: the "Cross" set and the "Dot" set (Fig.1) [7]. The watermark bits are firstly embedded in the "Cross" set, and then embedded in the "Dot" set.

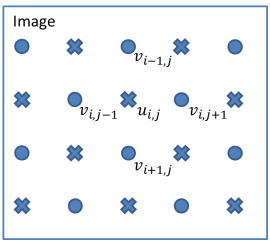


Fig. 1. "Cross" set and "Dot" set

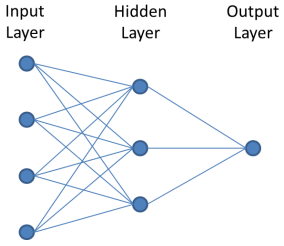


Fig. 2. Structure of neural network

2.1 Prediction Based on Classified Neural Network

During the "Cross" embedding, the "Cross" set is used for embedding data while the "Dot" set works as the reference signals. And vice versa. The center

pixel of a cell is predicted by the four neighboring pixels. In this paper, neural network is used to predict the pixel value considering the global feature. Since four pixels in the neighboring context are utilized to calculate the prediction value, the structure of the neural network is shown in Fig.2. The input layer has four neurons and the hidden layer is created with three neurons. The output layer has one neuron which refers to the central pixel value.

The corresponding weights and parameters can be determined by training. Considering the global influence and generalization, a great number of pixel cells from the original image are input to the neural network to obtain the models of prediction. Based on the variance of a central pixel's context, a pixel cell is assigned as a smooth one or a rough one. Subsequently, the pixel cells in smooth area are gathered to train a "smooth" neural network. And the pixel cells in rough area are used to obtain a "rough" neural network. Thus, the prediction value $u'_{i,j}$ is deduced using smooth neural network or rough neural network.

$$u'_{i,j} = \begin{cases} \lfloor smoothPredict(v_{i,j-1}, v_{i-1,j}, v_{i,j+1}, v_{i+1,j}) \rfloor \\ \quad \text{if } var_{i,j} < T_{var} \\ \lfloor roughPredict(v_{i,j-1}, v_{i-1,j}, v_{i,j+1}, v_{i+1,j}) \rfloor \\ \quad \text{if } var_{i,j} \geq T_{var} \end{cases} \tag{1}$$

where, $\lfloor . \rfloor$ is a floor function. $smoothPredict(\cdot)$ is the predictor for pixels in smooth region, and $roughPredict(\cdot)$ is the predictor for pixels in rough region. $var_{i,j}$ means the variance of the context of $u_{i,j}$, which is restricted by T_{var}.

2.2 Basic Data Embedding and Extraction

The combination of difference expansion and histogram shifting method proposed by [3] is also used in this paper.

If the prediction error $e_{i,j} = u_{i,j} - u'_{i,j}$ is within the region $[T_n, T_p]$, $e_{i,j}$ is expanded to $E_{i,j} = 2 \times e_{i,j} + b$. T_n is the negative threshold and T_p is the positive threshold. Otherwise, if the prediction error belongs to the region $(-\infty, T_n) \cup (T_p, \infty)$, the pixel does not carry any data and the prediction error is simply shifted as: $E_{i,j} = e_{i,j} + T_p + 1$, if $e_{i,j} > T_p$ and $T_p \geq 0$; $E_{i,j} = e_{i,j} + T_n$, if $e_{i,j} < T_n$ and $T_n < 0$. That is,

$$E_{i,j} = \begin{cases} 2 \times e_{i,j} + b & \text{if } e_{i,j} \in [T_n, T_p] \\ e_{i,j} + T_p + 1 & \text{if } e_{i,j} > T_p \text{ and } T_p \geq 0 \\ e_{i,j} + T_n & \text{if } e_{i,j} < T_n \text{ and } T_n < 0 \end{cases} \tag{2}$$

The watermarked value is computed by $U_{i,j} = u'_{i,j} + E_{i,j}$.

During extraction, if $E_{i,j} \in [2T_n, 2T_p + 1]$, $b = E_{i,j} \bmod 2$, and $e_{i,j} = \lfloor E_{i,j}/2 \rfloor$; if $E_{i,j} > 2T_p + 1$, $e_{i,j} = E_{i,j} - T_p - 1$; if $E_{i,j} < 2T_n$, $e_{i,j} = E_{i,j} - T_n$. That is,

$$e_{i,j} = \begin{cases} \lfloor E_{i,j}/2 \rfloor & \text{if } E_{i,j} \in [2T_n, 2T_p + 1] \\ E_{i,j} - T_p - 1 & \text{if } E_{i,j} > 2T_p + 1 \\ E_{i,j} - T_n & \text{if } E_{i,j} < 2T_n \end{cases} \tag{3}$$

Then, $u_{i,j} = u'_{i,j} + e_{i,j}$.

2.3 Retesting Strategy Using Parity Property

To reduce the length of location map, two-pass-testing was proposed [7]. If a pixel can be modified twice based on Eq.(2), it belongs to Class A; if the pixel is modifiable only once due to overflow or underflow during the second embedding test, the pixel belongs to Class B; if the pixel cannot be modified even once, it belongs to Class C. During the testing process, bit "1" is used as an embedding bit for positive prediction errors, and bit "0" is for negative prediction errors. The positions of Class B and Class C are marked in a location map, which is also embedded with the payload.

In the decode phase, use once-embedding-test to distinguish Class A, and Class B(or Class C). And further discriminate Class B and Class C using the location map. As mentioned in [7], during extraction phase, some pixel cells belonging to Class B will be misclassified to Class A if the actually embedded data equals "0" for positive prediction errors or "1" for negative ones. The case is caused because that the real embedded bit does not coincide with the testing bit. The shiftable pixel cells in Class B are shifted in both embedding phase and extraction phase, so this part of pixels can be correctly identified using once-embedding-test during the extraction procedure. As for the expandable pixel cells in Class B, bit "1" is used to test the overflow for positive prediction errors. When the to-be-embedded bit is "0", some pixels do not exceed 255 even undergoing the second embedding test.

To avoid the influence of the misclassification, Sachnev's method sacrifices the capacity of Class B. The fixed information is embedded in the expandable pixel cells in Class B, that is, bit "1" is always embedded in all the positive prediction errors and bit "0" for all the negative prediction errors. However, many images may contain a certain number of pixels of Class B. Rational use of Class B can increase the capacity of the algorithm. We utilize the parity characteristic and retesting strategy to activate the capacity of Class B.

After once-embedding-test during the extraction phase, the pixel cells are assigned into two parts: Part One contains the cells without overflow or underflow, and Part Two contains the overflow or underflow cells. As a result, Part One is the set consisting of Class A and partial Class B, while Part Two is the set containing Class C and part of Class B. It is obvious that the elements of Class B which are attributed in Part One are problem pixel cells. Since they will cause wrong localization in the location map, the problematic pixel cells should be identified further.

For the cells in Part One, a retesting detection is designed to distinguish the ambiguous cells belonging to Class B. As for the expandable pixel cells, $U_{i,j} = u'_{i,j} + E_{i,j} = u'_{i,j} + 2e_{i,j} + b$. Thus, $U_{i,j} - u'_{i,j} = 2e_{i,j} + b$. Due to $2e_{i,j}$ is an even number, $b = LSB(U_{i,j} - u'_{i,j})$, here $LSB(x)$ means the Least Significant Bitplane of x. For the positive prediction errors, if $U_{i,j} - u'_{i,j}$ is an even number, the embedded bit is not consistent with the testing bit. Fortunately, the pixel values can be adjusted to fit the case to the two-pass-testing. It is obvious that the difference between embedding "1" and embedding "0" equals 1. Thus, add one to the pixel value $U_{i,j}$ and retest the corresponding prediction error using

the testing bit "1". For the negative prediction errors, if $U_{i,j} - u'_{i,j}$ is an odd number, subtract one from the pixel value $U_{i,j}$ and retest the corresponding prediction error using the testing bit "0". If the retesting result shows the pixel cell is overflow or underflow, it belongs to Part Two. Otherwise, it still belongs to Part One. After the retesting, Part One only contains Class A, and Part Two contains Class B and Class C. Further classification is conducted to distinguish Class B and Class C with the help of the location map. The pseudo-code below describes the retesting process.

Algorithm 1. Retesting Strategy During Extraction

```
Embedding Test to get Ū
if Ū is overflow or underflow then
  U ∈ Part Two
else
  U ∈ Part One
  For Expanded Pixel Cells in Part One
  if U − u′ is even then
    U ⇐ U + 1
    Retesting using bit "1"
  end if
  if U − u′ is odd then
    U ⇐ U − 1
    Retesting using bit "0"
  end if
  if Ū is overflow or underflow then
    U ∈ Part Two
  end if
end if
```

3 Encoder and Decoder

3.1 Data Embedding

We first embed data in "Cross" set, then embed in "Dot" set. To recover data, threshold values T_n(7 bits), T_p(7 bits), payload size $|P_{cross}|$(18bits) or payload size $|P_{dot}|$(18bits), length of location map (18bits), and the position of the last processed prediction error (18bits) should be known first. We will embed these 68 bits into the first 68 pixel values' LSB. In addition, the parameters of the neural network are vital to data embedding and extraction. The structure of the neural network we used possesses 4 inputs, 3 hidden layers, and 1 output. There are 15 weights and 4 biases to constitute 19 parameters. Due to there are two neural networks involved in the prediction phase, 38 parameters are necessarily stored in the image. If 15 bits are used to record one parameter, 570 bits in total are used to determine the specific neural networks. The parameters of the neural network are embedded once, that is, the parameters will be embedded in the "Dot" set. In result, in "Cross" embedding, only 68 auxiliary bits need

to be embedded, while in "Dot" embedding, 638 auxiliary bits including the neural network parameters need to be stored in the image. The auxiliary bits are embedded in the front pixel values by using simple LSB replacement method. Accordingly, the original LSB values should be recorded with the payload.

The embedding method is designed as follows, taking "Cross" set for example:

Step 1: Calculate the prediction errors. For each pixel $u_{i,j}$, compute the prediction value and the corresponding prediction error $e_{i,j}$ based on the smooth neural network or the rough neural network.

Step 2: Sort the prediction errors. For each pixel $u_{i,j}$, compute the variance $Var_{i,j}$ of four neighbor pixels, which is used as the sorting parameter. Skip the first 68 pixels which will be used to carry the auxiliary bits. Sort the pixel cells according to the ascendingly sorted $\{Var_{i,j}\}$ to produce a sorted row of prediction errors e_{sort}.

Step 3: Determine the threshold. According to the two-pass-testing, all pixels are classified in one of classes A, B and C. Although the shiftable pixels can be modified, they cannot carry watermark bits. Only can the expandable pixels in Class A and Class B be capable of carrying data. Let set of expandable pixels in class A be EA. Let set of expandable pixels in class B be EB.

In the sorted prediction errors e_{sort}, create the location map L. If a pixel belongs to Class B, the corresponding element in the location map is marked as "0"; while if the pixel belongs to Class C, it is marked as "1" in the location map. If $|P_{cross}| \leq |EA| + |EB| - |L| - 68$ and $|EA| \geq |L|$ are both satisfied, the to-be-embedded bits can be successfully embedded. Here, $|.|$ means the cardinality of the set. Otherwise, increase the threshold T_p or decrease T_n, and repeat Step.3.

Step 4: Embed data. The location map L, the true payload P_{cross}, and the first 68 LSBs are embedded in the "Cross" part of the image by using the embedding method described in section 2.2. The location map L is firstly embedded in Class A. The elements belonging to Class A and Class B are all used to improve the capacity. If the last to-be-embedded bit is processed, the position of the last processed element is recorded, which implies how many prediction errors have been used. Then, use the auxiliary data to modify the LSB values of the first 68 pixels by simple binary replacement.

After 4 steps, the "Cross" embedding process is finished. The "Dot" embedding scheme uses the modified pixels from the "Cross" set for computing the predicted values. The original pixels from the "Dot" set are used for embedding data, and the embedding procedure is similar to the "Cross" embedding. Only difference relies on the size of the auxiliary information because the parameters of neural network are embedded in the "Dot" part. After the "Dot" embedding, the watermarked image is obtained.

3.2 Data Extraction

During extraction phase, data in "Dot" set are extracted firstly, then the "Cross" set will be decoded. The parameters of neural network can be read from the LSBs of "Dot" set, which are used for pixel prediction both in "Cross" set and "Dot" set. We only describe the "Cross" decoding method in details below,

Step1: Recover the thresholds and calculate the prediction values. Read LSB values from the first 68 pixels to recover the values of T_n, T_p, payload size $|P_{cross}|$, and the position of the last processed prediction error. For each pixel value $U_{i,j}$, compute the prediction value based on the corresponding neural network. Afterwards, the prediction error $E_{i,j}$ is obtained.

Step2: Sort the prediction errors. Skip the first 68 pixels. Sort the pixels according to $\{Var_{i,j}\}$ to get a set of sorted prediction errors E_{sort}.

Step3: Extract the watermark. Test every pixel cell to classify it into Part One or Part Two. In detail, bit "1" is used as a testing watermark and embedded in the positive prediction errors. While bit "0" is embedded in the negative prediction errors. If the embedded pixel intensity exceeds the pixel range [0,255], it belongs to Part Two. If the pixel value still stays in the pixel range, it belongs to Part One which implies that the pixel may come from Class A. Because some pixel cells belonging to Class B are misclassified to Class A, further testing is conducted to recognize the problem elements in Part One. For the expandable pixels in Part One, if $U_{i,j} - u'_{i,j}$ is even and positive, add one to $U_{i,j}$ and retest the corresponding pixel using the testing bit "1". If $U_{i,j} - u'_{i,j}$ is odd and negative, subtract one from $U_{i,j}$ and retest the corresponding pixel using the testing bit "0". If the retesting result shows the pixel is overflow or underflow, it belongs to Part Two. Otherwise, it still belongs to Part One. After the retesting, Part One is Class A, and Part Two contains Class B and Class C. Extract location map from Class A firstly. Further classification is conducted to distinguish Class B and Class C based on the location map. Then, extract data from Class A and Class B, meanwhile recover the original prediction errors using the method in section 2.2. The extraction procedure is only applied to the embedded prediction errors according to the position of the last processed element. The extracted data is the cascading of the true payload and the 68 LSBs.

Step4: Restore the original image. Computer the original pixel values based on $u_{i,j} = u'_{i,j} + e_{i,j}$.

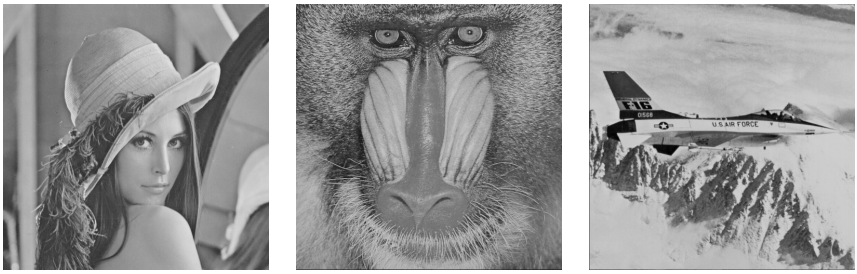
Step5: Recover the rest pixels. Replace the first 68 LSB values of the pixels with the extracted 68 LSBs.

When the "Dot" and "Cross" decoding are both finished, the entire watermark is obtained and the original image is restored.

4 Experimental Results

Several 8-bit gray images with size 512×512 are used in the experiments. Fig.3 shows the original images: "Lena", "Baboon" and "Plane".

The maximum capacity under a certain threshold region is compared between the proposed algorithm and the one in [7]. Take "Lena" image for example, and use the same threshold region for both "Cross" set and "Dot" set. Table 1 gives their maximum capacities. Our method can get higher capacity.



(a) Lena (b) Baboon (c) Plane

Fig. 3. The original gray images

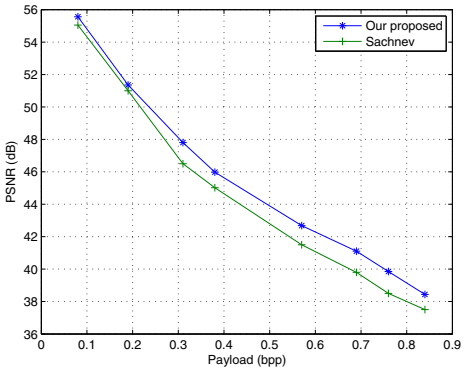
Table 1. Maximum Capacity Comparisons for "Lena"

threshold [Tn Tp]	max capacity of our(bits)	max capacity of [7](bits)
[-2 2]	158429	157355
[-3 3]	191848	190725
[-5 5]	227781	226567
[-7 7]	243237	242167
[-9 9]	250553	249686

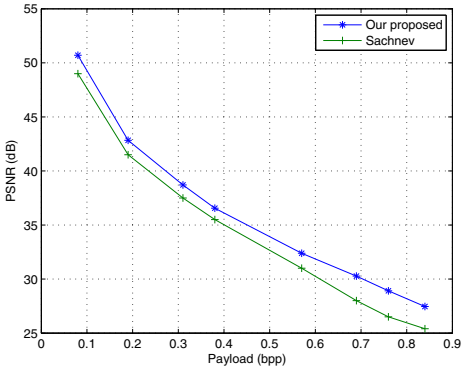
Fig.4 shows the performances of Capacity vs. Visual quality in terms of payload and PSNR(Peak Signal-to-Noise Ratio). The horizontal axis represents the capacity in terms of bpp(bits per pixel). The vertical axis represents the corresponding PSNR. The results show that our method have both high visual quality and high capacity. Compared with [7], our method can achieve better results for "Lena" and "Baboon", and get comparable result for "Plane". The reason is the neural network may not be trained well.

5 Conclusions

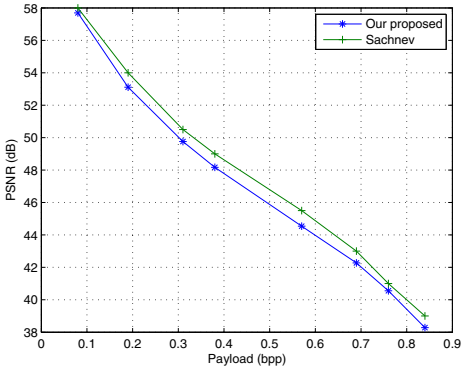
In this paper, we propose a high capacity image reversible watermarking based on classified neural network and retesting strategy. Considering the global feature, two neural networks are designed for smooth pixel prediction and rough pixel prediction, respectively. The parameters of the neural network are also stored in the image. Furthermore, a retesting strategy utilizing the parity detection activates the capacity of the ambiguous pixel cells. As a result, the capacity is increased. The watermark is embedded in the prediction errors combining expansion and shifting methods. Experimental results show that the proposed algorithm can obtain higher capacity and preserve good visual quality.



(a) Result for "Lena"



(b) Result for "Baboon"



(c) Result for "Plane"

Fig. 4. Capacity vs. PSNR for testing images

References

1. Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible Data Hiding. IEEE Trans. on Circuits Syst. Video Technol. 3, 354–362 (2006)
2. Tian, J.: Reversible Data Embedding Using a Difference Expansion. IEEE Trans. on Circuits Syst. Video Technol. 8, 890–896 (2003)
3. Thodi, D.M., Rodriguez, J.J.: Expansion Embedding Techniques for Reversible Watermarking. IEEE Trans. on Image Process. 3, 721–730 (2007)
4. Tsai, W.L., Yeh, C.M., Chang, C.C.: Reversible Data Hiding based on Histogram Modification of Pixel Differences. IEEE Trans. on Circuits Syst. Video Technol. 6, 906–910 (2009)
5. Tsai, P.Y., Hu, C., Yeh, H.L.: Reversible Image Hiding Scheme Using Predictive Coding and Histogram Shifting. IEEE Signal Process. Mag. 6, 1129–1143 (2009)
6. Luo, L.Z., Chen, N., Zeng, X., Xiong, Z.: Reversible Image Watermarking Using Interpolation Technique. IEEE Trans. on Inf. Forensics and Security 1, 187–193 (2010)
7. Sachnev, V., Kim, H.J., Nam, J., Shi, Y.Q., Suresh, S.: Reversible Watermarking Algorithm Using Sorting and Prediction. IEEE Trans. on Circuits Syst. Video Technol. 7, 989–999 (2009)
8. Coltuc, D.: Improved Embedding for Prediction-Based Reversible Watermarking. IEEE Trans. on Inf. Forensics and Security 3, 873–882 (2011)